University of Ljubljana
Faculty of Computer and Information Science

Ivana Kostadinovska

# Cloud security - An approach with modern cryptographic solutions

masters thesis

the 2nd cycle masters study programme
computer and information science

Supervisor: prof. dr. Denis Trček

Ljubljana, 2016

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Ivana Kostadinovska

# Varnost v oblaku - Pristop s sodobnimi kriptografskimi rešitvami

magistrsko delo

magistrski program druge stopnje
računalništvo in informatika

Mentor: prof. dr. Denis Trček

Ljubljana, 2016

# Declaration of Masters Thesis authorship

I, the undersigned Ivana Kostadinovska am the author of the Masters Thesis entitled:

*Cloud security - An approach with modern cryptographic solutions*

With my signature, I declare that:

- the submitted Thesis is my own unaided work under the supervision of prof. dr. Denis Trček
- all electronic forms of the Masters Thesis, title (Slovenian, English), abstract (Slovenian, English) and keywords (Slovenian, English) are identical to the printed form of the Masters Thesis,
- I agree with the publication of the electronic form of the Masters Thesis in the collection "Dela FRI".

In Ljubljana, 22. March 2016 Author's signature:

# Acknowledgments

*I would like to express my sincere gratitude to my advisor Prof. Dr. Denis Trček for guiding me through the writing of this thesis, for his patience, and immense knowledge. His guidance and ideas helped me improved this thesis. I would also like to thank my parents and my brother for providing me with unfailing support and continuous encouragement throughout my years of study. Last but not the least, I would like to thank my friends for their support and help.*

*Ivana Kostadinovska, 2016*

To my family.

# Contents

# Povzetek

Izraz "računalništvo v oblaku" je zaradi potenciala spremembe trenutne računalniške industrije v središču pozornosti računalničarjev. Žal pa še vedno obstajajo ovire, ki jih je treba razrešiti, saj so v računalništvu, osnovanem v oblakih, varnostni aspekti še vedno osrednji problem.

Cilj našega dela je identificirati glavne probleme varnosti v zvezi z računalništvom v oblaku in predstaviti plasti za varne oblake. Naše raziskave se osredotočajo tudi na podatkovne in shrambene varnostne plasti. Ugotovili smo, da zaščita podatkov v oblaku leži v kriptografiji oblaka. Ta magistrska naloga prikaže nove kriptografske tehnike, uporabljene za zaščito in procesiranje zakodiranih podatkov v oddaljeni oblačni shrambi.

V nalogi predlagamo kriptografsko shemo, ki uporabi skeniranje prstnega odtisa za preverjanje verodostojnosti uporabnika in AES tehnike 128/192/256 bitnega ključa za šifriranje in dešifriranje uporabnikovih podatkov. AES omogoča večjo podatkovno varnost v primerjavi z drugimi tehnikami šifriranja, kot na primer DES in Blowfish. Našo shemo smo uporabili v aplikaciji DropboxCrypt. DropBoxCrypt je aplikacija za šifriranje in dešifriranje podatkov, razvita za mobilne naprave Android, ki jo lahko uporabljamo za brskanje, pridobivanje in odpiranje šifriranih podatkov, shranjenih v oblačnih shrambah.

## Ključne besede

*računalništvo v oblaku, varnost v oblaku, kriptografija*

# Abstract

The term “cloud computing” has been in the spotlights of IT specialists due to its potential of transforming computer industry. Unfortunately, there are still some challenges to be resolved and the security aspects in the cloud based computing environment remain at the core of interest.

The goal of our work is to identify the main security issues of cloud computing and to present approaches to secure clouds. Our research also focuses on data and storage security layers. As a result, we found out that the protection of cloud data lies in cloud cryptography. Thus, this thesis reviews the new cryptographic techniques used to protect and process encrypted data in a remote cloud storage.

In this thesis we are proposing a cryptographic scheme which uses fingerprint scanning for user authentication and AES technique of 128/192/256 bit cipher key for encryption and decryption of user’s data. AES provides higher data security compared to other encryption techniques like DES and Blowfish. Our scheme is used in DropBoxCrypt application. DropBoxCrypt is a data encryption-decryption application developed for Android mobile devices which can be used for browsing, exporting and opening encrypted data stored in cloud storage.

## Keywords

*cloud computing, cloud security, cloud cryptography*

# Chapter 1

# Cloud computing concepts

## 1.1 Introduction

Cloud computing is in the spotlight of computer industry today. With its new aspects and capabilities that have been proclaimed, cloud computing is a rapidly evolving model. Cloud may be the next evolution in IT history and is radically changing the way an enterprise manages its informatics systems.

There are plenty of definitions what "cloud computing" is. According to NIST [1], cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

IBM says that cloud computing, often referred to as simply "the cloud" is a delivery of on-demand computing resources — everything from applications to data centres — over the Internet on a pay-for-use basis [2].

Gartner defines cloud computing as a style of computing in which scalable and elastic IT-enabled capabilities are delivered as a service using Internet technologies [3].

Cloud computing is based on five attributes: multitenancy (shared resources), massive scalability, elasticity, pay as you go, and self-provisioning

of resources [4].

In simple words, it is the practice of using a network of remote servers hosted on the Internet to store, manage, and process data, rather than a local server or a personal computer - it allows application software to be operated using internet-enabled devices [5].

## 1.2 Main features

Although there are plenty of definitions about cloud computing, we can separate a few main features. NIST [1] gave a definition of cloud computing, which stated that the latter includes five essential characteristics:


**Figure 1.1:** Five essential characteristics of cloud computing [6]

**On-demand self-service:** Users are able to manage cloud services, such as provision of computing power, storage, networks and software, using only a simple online control panel. Those utilities are available whenever they are required, without human interaction with each service provider.

**Broad network access:** Cloud services are available from every part of the world using standard network protocols and are accessible from every device with internet connection (workstations, laptops, tablets and smart phones).

**Resource pooling:** It allows cloud providers to pool large-scale IT resources to serve multiple consumers with services at the same time. Providers create a sense of immediately available resources and location independence. On other hand, consumers have no control over the exact location of resources, unless they specify the location on a higher level of abstraction such as country or datacenter. The ability of serving one instance of a program to different consumers is referred to as multitenancy. Multitenancy model relies on the use of virtualisation and that way, according to demands, IT resources can be dynamically assigned and reassigned [7].

**Rapid elasticity:** Ability to provide scalable services. Rapid elasticity allows consumers to automatically request different types of services or additional space, which means that resources are scaled up and down rapidly as needed.

**Measured service:** Or pay as you go principle, is a way in which a provider measures the effective use of services and resources of a consumer.

## 1.3 The cloud delivery model

Service oriented architecture (SOA) is a modern model, which is a base for cloud computing. It focuses on delivering integrated services to consumer, using different functions and approaches. With different combinations of these functionalities, providers can offer a few ways of using the benefits of the cloud. According to that, there are three fundamental models: software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS). This concept is known as SPI (Software-Platform-Infrastructure) service delivery model [8] [9] [1] [4] [10].


**Figure 1.2:** Cloud service delivery methods [6]

### 1.3.1 Software as a Service (SaaS)

SaaS offers complete and finished application on demand and represents the ability provided to the consumer to use one or more computer resources running on a cloud infrastructure. SaaS operates in the following manner: a single software instance runs in a cloud and supports multiple end consumers at the same time. The consumer has access to the service through any authorized device across the Internet, with no need of installing and running the application on that device. The consumer rents the software on a pay-per-use model or has free limited use for some services.

The number of tenants supported by the application is the main difference between traditional software model and SaaS model. Traditional software represents a single-tenant model, it is installed on a server and runs only for a single consumer. On the other hand, the multi-tenant architecture model used by SaaS allows one physical hardware infrastructure to be shared among many different consumers, and it will be unique for each of them [4].

Key benefits of using the SaaS model are [4]:

- The cost of software license, hardware, operations and maintenance is reduced by hosting to a third party.
- SaaS centralized control allows vendors to control and limit the use.
- The majority of SaaS applications are delivered with the Web as infrastructure. Some SaaS vendors provide their own application interface,

whereas other SaaS apps can be accessed via web browser.

- SaaS deployment does not require any specific hardware or software.
- Vendors are responsible for the management of SaaS applications.

In this model, consumer cannot manage cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, except some limited user configuration settings.

Most widely used examples of SaaS include Salesforce CRM, Google Docs, Facebook, Twitter, Yahoo Email, Gmail, MobileMe, Zoho, etc.

### 1.3.2 Platform as a Service (PaaS)

PaaS gives a consumer an operating system and programming environment and offers its usage for every phase of software development and testing. PaaS enables a software deployment model in which consumers are able to develop and deploy their own application to the cloud infrastructure, using programming languages, libraries, services, and tools supported by provider. Provider is paid for providing these platforms and distribution services.

PaaS services represent a development platform composed of predefined building blocks, used by consumers to create their own applications. Development tools are hosted in the cloud and there is no need for additional tools on developer's computer or any system administration skills for deployment. Due to low cost and complexity of buying and setting servers up, PaaS systems are widely used by start-up companies and self-employed developers [4].

There are a few differences when compared to traditional development environments. Multi-tenant development tools in clouds support multiple users with multiple active projects, while traditional tools are designed for only one user. Multi-tenant deployment architecture offers application and data scalability and supports runtime monitoring [4].

In this model, consumer cannot manage cloud infrastructure including network, servers, operating systems, or storage, but has control over deployed

applications. PaaS can reduce the cost of buying, housing and managing software platforms.

Examples of such platforms are Google App Engine, Microsoft Azure, IBM SmartCloud Application Services, Amazon Web Services, etc.

### 1.3.3 Infrastructure as a Service (IaaS)

IaaS provides use of basic computing infrastructure of servers, software and network equipment and other fundamental computing resources, which are presented in an on-demand service form. Consumers have direct access to storage, processing and other computing resources throughout the network, where they are able to deploy and run appropriate software [10]. On the other hand, they do not have insight into infrastructure details, including location, data partitioning, security, backup and scaling.

The novelties brought by cloud hosting have enabled that pay-per-use model is offered and that services are scaled. On provider's side, the offered infrastructure can increase or decrease its capacity, depending on consumer's demands. Except for providing application hosting, some IaaS providers can support a wide range of additional services, such as application development, support and enhancements [4].

The IaaS model is similar to utility computing [4], where computing services are provided on-demand, just like utilities [11]. Consumer cannot manage cloud infrastructure but has control over storage, operating systems and deployed applications. With IaaS we can avoid buying, housing and managing the basic hardware and software infrastructure [1].

Commercial examples include Amazon Elastic Compute Cloud (EC2), Terremark Enterprise Cloud, Rackspace, Joyent, IBM Computing on Demand, etc.

## 1.4 Cloud deployment models

Depending on requirements, there are a few ways of deploying cloud services [1]: private cloud, community cloud, public cloud, and hybrid cloud. Each of them offers some additional benefits.

### 1.4.1 Private cloud

Private cloud infrastructure is built for exclusive use of consumers of a specific organization, where the enterprise computing architecture is protected by a firewall. An organization has control over security, data and quality of service. The cloud may be owned, managed, and operated by an organization or by a third party and may be deployed in its own datacenter [10].

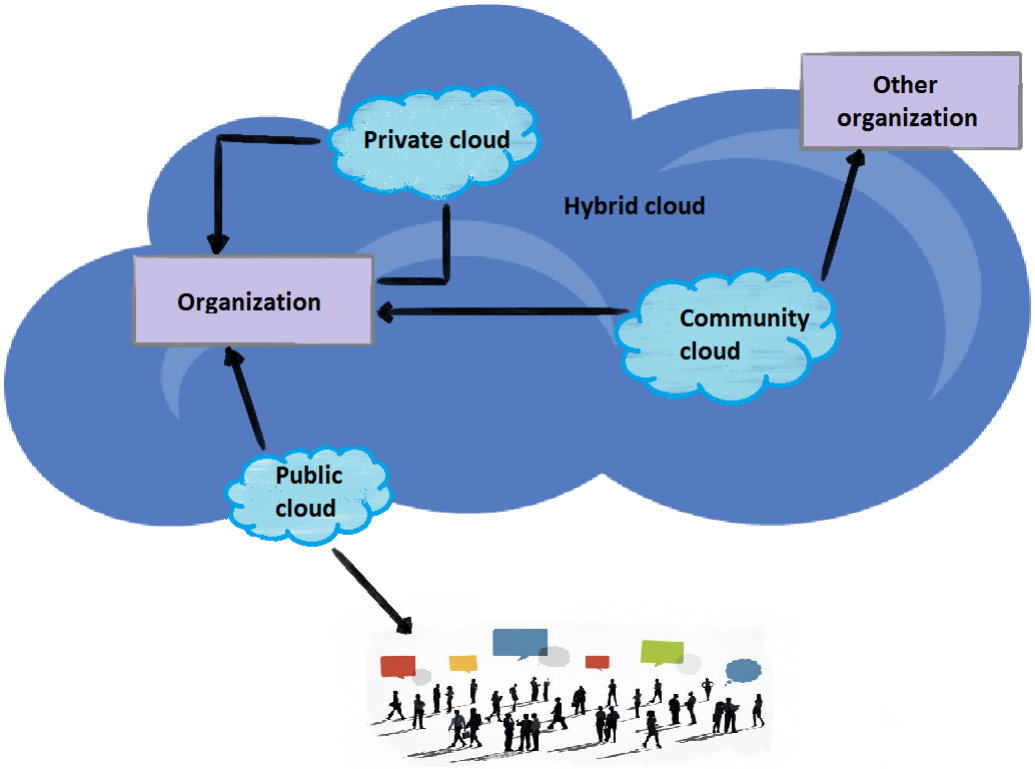


**Figure 1.3:** Cloud deployment models

Private cloud is dedicated to a single organizational tenant. An organization must buy, build and manage the cloud with its pitfalls, governance and reliability concerns. The customer is responsible for cloud operation and should have a high degree of control of physical and logical security aspects of cloud infrastructure [4].

### 1.4.2 Community cloud

In community cloud, the infrastructure is used by a specific community of consumers from organizations that have similar requirements (compliance, security, policy) or share the same infrastructure (government, healthcare, finance). The cloud may be owned, managed, and operated by one or more of the organizations in the community, a combination of them or by a third party. The costs are spread over fewer consumers [1].

### 1.4.3 Public cloud

Infrastructure in public clouds is open for general public on the Internet, where everyone has free access to use applications and storage of service provider. Resources and services are available over the Internet as web applications or web services [4]. Public clouds reduce consumer risk and cost by providing a flexible entry in cloud infrastructure. They may be owned, managed, and operated by a business or government organization, or a combination of them, from one or more data centres. Public cloud services may be free or offered through a pay-per-usage model [1]. Security management and cloud service operations have obligations towards vendors, because a consumer does not have any control and oversight over physical and logical security aspects [4].

### 1.4.4 Hybrid cloud

Hybrid cloud is a composition of two or more clouds (private, community or public) that remain unique entities, but are bound together by standardized

or proprietary technology. It offers the benefits of multiple deployment models [1]. With the help of this cloud, organizations can use public cloud for less sensitive applications, and a private cloud for maintenance of core and sensitive applications and data [4].

## 1.5 Moving to the cloud

Although cloud computing adoption brings many benefits, there are security issues that may present possible risk. If it's important to move some critical applications to the cloud, it's recommended to apply additional security controls. When adopting the cloud, it is crucial to have a thorough understanding of how to integrate cloud solutions into existing architecture. Having in mind that there is such variety of cloud deployment options and cloud delivery models, it's impossible to cover all circumstances with a single list of security controls. However, a risk-based approach to moving to the cloud may help with initial cloud risks and security decisions. For that purpose, CSA [12] recommends following a quick simple framework for evaluating the tolerance for moving an asset to various cloud computing models [13]:

**Identifying the asset for cloud deployment**

Assets that can be supported may be data or applications, functions and processes. They don't have to be located on the same location or present all parts of the function; some parts can be hosted in their own datacentres and some parts in the cloud. In this step, it is important to determine exactly which data or function is being considered for the cloud.

**Evaluating the asset**

After identifying the asset, we need to determine how important the data or function are for a consumer. It is sufficient to answer a few simple questions, for example, how would one be harmed if the asset became widely public and distributed, if the process was manipulated

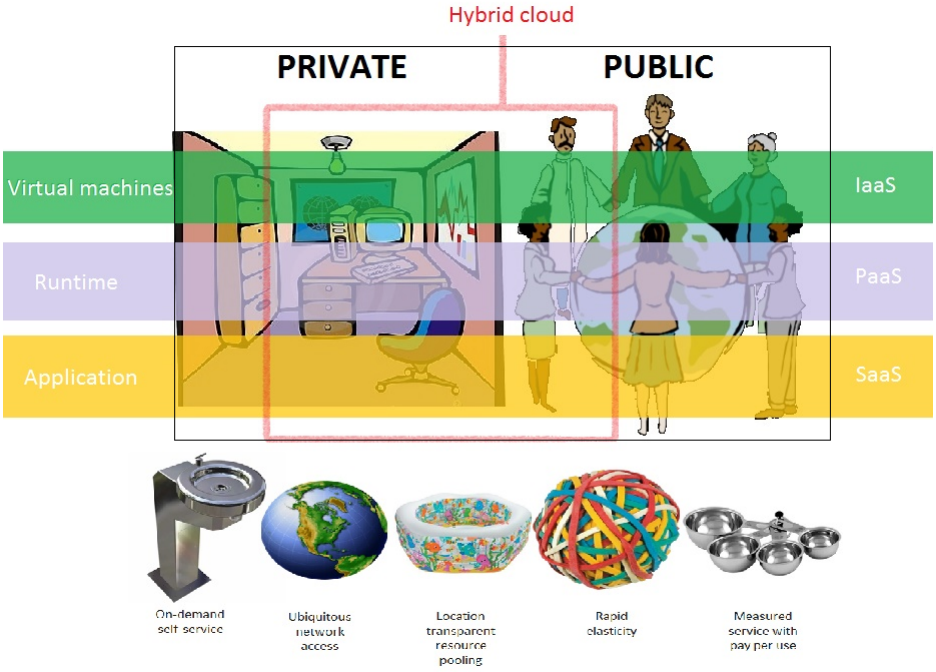


**Figure 1.4:** 5-3-2 principle of cloud computing

by an outsider or if the asset was unavailable for a period of time. There is no need for detailed valuation, we only need the essential assessment of confidentiality, integrity and availability of the asset.

**Mapping the asset to potential cloud deployment models**

It is of great importance to determine which deployment and hosting model fits security and risk requirements for the given asset. We should know the strength of security that is required and our comfort levels for transitioning.

**Evaluating the potential cloud service models and providers**

This step is focused on the degree of control needed at each SPI tier to implement any required risk management.

**Sketching the potential data flow**

If a specific deployment option is being evaluated, data flow between

an organization, a cloud service and users needs to be mapped out. Before making a final decision, it's essential to understand whether and in what manner data can move in and out of the cloud.

### 1.5.1 Summary

Before they move to the cloud, customers should understand the sensibility of the information they want to store, the risks and which combinations of deployment and service models are acceptable for them. Needed security levels are different for various types of data. Low-value data do not require high level of security controls and many of the recommendations can be skipped, such as complex encryption schemes. Not all cloud deployments need every possible security and risk control. It is left for customers to decide which level of security they will apply [13].

# Chapter 2

# Cloud security

Cloud computing is at an advantage compared to the traditional IT models. However, security concerns remain a major barrier for its adoption. A survey from IDCI in 2009 shows that 74% of IT managers and CIOs believed that cloud computing security issues are the primary challenge that hinders them from using cloud computing services [14]. Surveys show that the top issue organizations have when considering the adoption of public cloud-based computing services is "security and privacy" [15]. Additionally, more than 70% of CTOs believed that the primary reason not to use cloud computing services is that there are data security and privacy concerns [15].

Major cloud computing vendors had successively reported several accidents in 2008 and 2009. Amazon's Simple Storage Service was interrupted twice in February [16] and July 2008 [17]. In March 2009 [18], security vulnerabilities in Google Docs even led to serious leakage of private user information. Google Gmail was experiencing a global failure for more than two hours in February 2009 [19]. There was serious security vulnerability in VMware virtualization software for Mac version in May 2009 [20]. Microsoft's Azure cloud computing platform also experienced a serious outage accident for about 22 hours in March 2009 [21]. Serious security incidents even lead to the collapse of cloud computing vendors. As administrators' misuse had led to the loss of 45% of user data, cloud storage vendor LinkUp

was forced to close in August 2008 [22].

Security control measures in cloud are similar to those in traditional IT environment. However, cloud computing may face different risks and challenges, because of multi-tenant characteristics, service delivery models and deployment models.

## 2.1 Cloud computing security issues

### 2.1.1 Cloud computing security

Cloud computing security is an evolving sub-domain of computer security, network security, and, more broadly, information security. It refers to a broad set of policies, technologies, and controls deployed to protect data, applications, and the associated infrastructure of cloud computing [23]. It represents a response to a familiar set of security challenges that manifest differently in the cloud. It contains a set of policies, technologies, and controls designed to protect data, infrastructure, and clients from an attack and enable regulatory compliance. Security is higher when layered at each level of the stack and integrated into a common management framework and thus providing protection whatever delivery model is used [24].

### 2.1.2 Security issues associated with the cloud

There are many security issues associated with cloud computing. Cloud Security Alliance (CSA) has identified thirteen domains of concerns over cloud computing security in its second version of Security Guidance [13] and fourteen domains in the third edition [25]. S. Subashini and V. Kavitha give an overview of cloud computing security issues related to cloud computing service delivery models (SPI model), and present a detailed analysis for each security issue in all aspects of infrastructure, including network level, host level and application level [26]. According to Gartner [27], users should ask vendors for seven specific safety issues: privileged user access, regulatory

compliance, data location, data segregation, recovery, investigative support and long-term viability. Forrester Research Inc. [28] has evaluated security and privacy practices of some of the major cloud providers in three major aspects: security and privacy, compliance, and legal and contractual issues. Mohamed Al Morsy, John Grundy and Ingo Müller explored cloud computing security issues from different perspectives, including security issues associated with cloud computing architecture, service delivery models, cloud characteristics and cloud stakeholders [29] [20].

### 2.1.3 Security issues according to Cloud Security Alliance

Cloud Security Alliance (CSA) has identified fourteen domains of concerns over cloud computing security, divided into two broad categories: governance and operations [25].

#### Domain 1: Cloud computing architectural framework

This domain provides a conceptual framework for Cloud Security Alliance's guidance. Here are some basic issues, necessary to ensure cloud computing security: terminology, architectural requirements and challenges for securing cloud applications and services, and a reference model that describes taxonomy of cloud services and architectures.

CSA defined cloud computing as [13]:

> Cloud computing ('cloud') is an evolving term that describes the development of many existing technologies and approaches to computing into something different. Cloud separates application and information resources from the underlying infrastructure, and mechanisms used to deliver them.

CSA focuses also on the architecture. From architectural perspective, cloud is both similar to and different from existing models of computing

and this thin line between them impacts the organizational, operational and technological approaches to data, network and information security practices. The main issue for understanding security risks is to understand the relationship between models in the cloud reference model diagram. IaaS is located at the top of the model diagram, meaning it represents the foundation of other models. By this model, the services inherit not only their capabilities, but also security issues and risks. According to provider/consumer security responsibilities, there are also differences between these levels. The lower provider goes into the reference model stack, the more responsible consumer becomes for implementing and managing security.

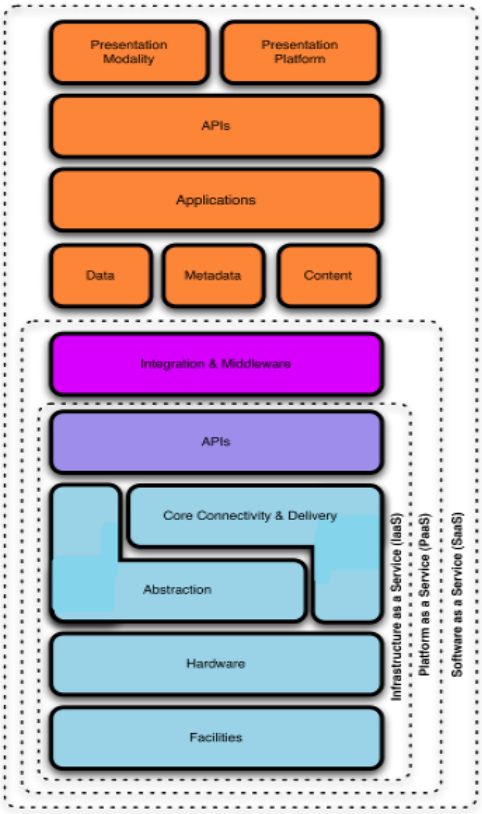


**Figure 2.1:** Cloud Reference Model [13]

The cloud security reference model concentrates on relationships between all cloud services and their security controls. It is of great importance to look into all possible cloud scenarios. Important is also whether the cloud is private or public, and to correlate that with physical location of the data, information on who used the cloud and who managed it and how. Examples of visualizing those relationships are shown in the next table of cloud computing deployment models and in the Jericho Cloud Cube Model [30].

The Cloud Cube Model illustrates the variations of cloud offerings, using four different dimensions [30]:

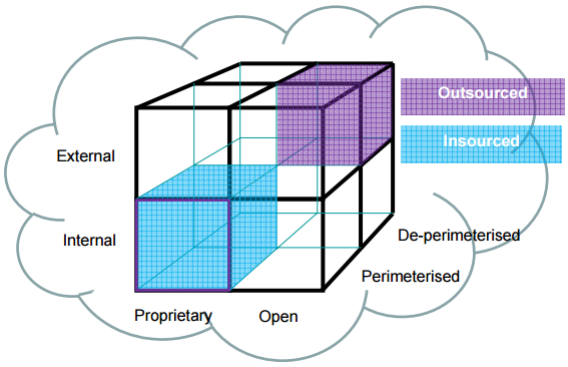


**Figure 2.2:** Cloud Cube Model [30]

**Internal (I) / External (E)** – dimension that defines physical location of the data – organization boundaries

**Proprietary (P) / Open (O)** – dimension that defines the state of ownership of the cloud technology, interfaces and services

**Perimeterised (Per) / De-perimeterised (D-p) Architectures** – dimension that describes if we operate inside our traditional IT perimeter or outside it

**Insourced / Outsourced** – dimension that explains who is running the cloud. Insourced means that the service is provided by our own staff, and outsourced means that the service is provided by a 3rd party

### Domain 2: Governance and enterprise risk management

This domain shows an organization's ability to govern and measure enterprise risk introduced by cloud computing. In order to support effective security governance and compliance, the main issues of the governance and enterprise risk management are the identification and implementation of the suitable organizational structures, processes and controls [13]. An effective governance plan must be formed from well-developed information security governance processes, as a part of organization's obligations. The enterprise

risk management is obliged to provide value for its partners, in a way that can measure, manage and mitigate uncertainty. It uses methods and processes to identify and analyse risk, and pick the appropriate risk response strategy, which may include: avoidance, reduction, sharing or insuring and accepting. The risk response strategy is also affected by the decision of information risk management, which aligned the answer with the risk tolerance of data owner [25].

There are some recommendations from CSA [12] about this domain. Main suggestions are for investing into increased scrutiny of security capabilities of the provider, including review of specific information security governance structures and processes by user organization and to be mandatory to establish collaborative governance structures and processes between customers and providers. Also, the issue of not having physical control over the infrastructure, makes risk management for cloud computing development more sensitive compared to the traditional model. For this reason, it is necessary to have Service Level Agreements provider documentation and contract requirements [13] [25].

#### Domain 3: Legal issues: contracts and electronic discovery

This domain provides general background on potential legal issues for using cloud computing services, such as protection requirements for information, regulatory requirements, privacy requirements, international laws, etc. The need for new legislation arises from the novelty brought by cloud computing paradigm, such as the time of service (on-demand and intermittent), the anonymity of identity of a service provider and the anonymity of the location of involved servers. When we talk about legal issues related to cloud computing, we need to consider them through functional, jurisdictional and contractual dimension. The functional dimension discovers which cloud computing services and functions have legal implications for contributors, and the jurisdictional dimension covers the way in which laws and regulations affect services, data assets and contributors. The managing of and addressing le-

gal issues is enabled by the contractual dimension, which includes contract structures, enforcement mechanisms, and terms and conditions [13].

Considering different locations of data throughout the world, cloud services may be subject to different laws according to the legislation of the country in which cloud servers are. The countries from the Asia Pacific region, have accepted data protection laws based on Privacy and Security Guidelines of the Organization for Economic Cooperation and Development (OECD) [31] and the Asia Pacific Economic Cooperation's (APEC) Privacy Framework [32]. In Europe, the Europe Economic Area (EEA) Member States have passed data protection laws tracked by the European Union (EU) Data Protection Directive of 1995 [33] and the ePrivacy Directive of 2002 (amended in 2009) [34]. Similar laws have been adopted by Morocco and Tunisia in Africa, and Israel and Dubai in the Middle East. North, Central and South America countries adopted data protection laws inspired mostly from the European model, except the United States of America, where there are more data protection laws and related regulations, contained in the US Federal Laws and US State Laws, such as GLBA[1] [35], HIPAA[2] [36], COPPA[3] [37], etc. [25].

The data transferred to the cloud must be protected by service provider on the same level it was protected in the hands of the owner. Thus, it is a legal requisite that a data owner and a cloud provider make a written agreement (contract) which will define the roles, expectations and allocations between the participants. These obligations must be attained due diligence (before execution of the contract) or security audits (during performance of the contract) [25].

[1]Gramm–Leach–Bliley Act
[2]Health Insurance Portability and Accountability Act
[3]Children's Online Privacy Protection Act

### Domain 4: Compliance and audit management

Finding how to maintain compliance and dealing with evaluation of the effects of cloud computing on compliance with internal policies, are the main goals for this domain. CSA advises the cloud customer to consider and understand the regulatory applicability for given cloud service, compliance responsibilities between the provider and the customer, provider ability to express compliance and its relationship with providers and auditors [13].

Compliance is the last pillar of the GRC[4] [38] model that we need to mention. Figure 2.3 shows the tight connection between governance, risk management and compliance for ensuring security in the cloud system. While the corporate governance gives balance between cloud participators and provides guidance and controls, and the enterprise risk management gives methods and processes to help making decisions, the compliance gives the adherence and awareness to company obligations. Compliance activities ensure that operations are fully aligned with governance processes and policies. On the other hand, the audit plan should provide a thorough testing and reviewing of risk management activities [25].

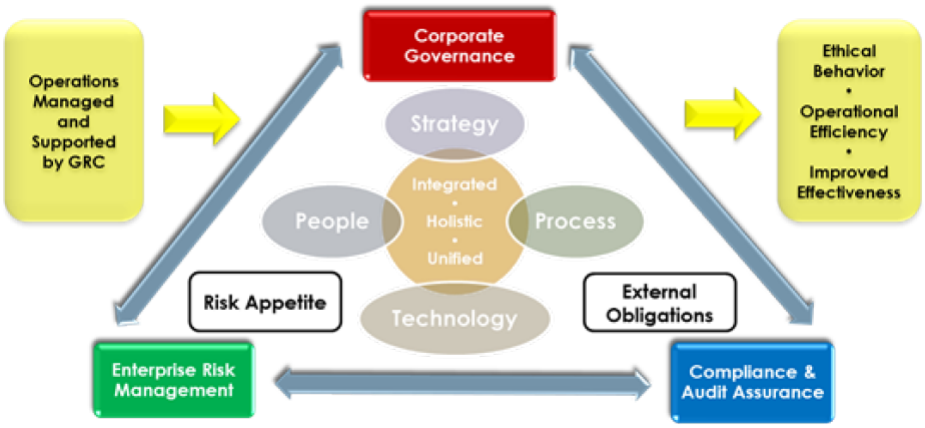


**Figure 2.3:** GRC model [31]

[4]Governance, risk management and compliance

### Domain 5: Information management and data security

This domain gives overview of the management of cloud data, the identification and control of data, and controls which can be used to deal with the loss of physical control when moving data to the cloud. Those processes and polices are included in the information management. For evaluating and defining cloud data security strategy, CSA proposes using Data Security Lifecycle [39]. According to the lifecycle, data go through six phases: create, store, use, share, archive and destroy. But this lifecycle doesn't include all we need for the process of managing information. We also need to know the logical and physical location of data, because DSL phases are running in different operating environments, and data are moving in, out of and between these environments. Also, it is important to consider who and how accesses to the data, and to identify functions and controls that can be performed [39].

Applying information governance is provided by data security with its specific controls and technologies. Data security includes detecting and preventing data migration to the cloud, protecting data moving to and within the cloud, and protecting data in the cloud [25].

### Domain 6: Portability and interoperability

This domain highlights the ability to move data and services from one provider to another. From a security perspective, it is required to maintain security controls consistency in the processes where the environment is changed [13].

CSA gives numerous recommendations in relation to interoperability, portability and different cloud models. For hardware, it is recommended to use virtualization whenever possible to remove many hardware level issues, and to use open virtualization formats such as OVF. Changing cloud provider does not always bring the same framework as previously used, so it is crucial to investigate the new API and determine the differences. Before using any cloud, it is a good practice that the customer chooses open and published platforms to ensure broadest support for interoperability. From storage viewpoint, the unstructured data need to be stored in an established

portable format and need to be assessed for encryption. On the other hand, portability recommendations include: service level, different architecture, security integration, authentication and identity management to operate across all components of the cloud, encryption keys to be escrowed locally and to ensure that the copies of the file metadata are securely removed [25].

**Domain 7: Traditional security, business continuity and disaster recovery**

Moving data to the cloud affects traditional operational processes and procedures used to implement security, business continuity and disaster recovery. The focus in this topic is to discuss possible security risks and the fact that some of them are unique for cloud computing [13].

When establishing traditional security functions in the cloud, must take into consideration the evaluation of physical security and security infrastructures. For physical security, it is recommended to check the physical location of the cloud service provider facility and to review the documentation that supports recovery operations, such as risk analysis, risk assessments, vulnerability assessments, business continuity plans, disaster recovery plans, physical and environmental security policy, etc. After that, the customer must make certain that the CSP[5] complies with global security and industry standards, such as ISO 27001 ISMS, TOGAF[6], SABSA[7], ITIL[8], COSO[9] or COBIT[10]. To secure cloud data from intruders accessing to facilities, the Four D's[11] of Perimeter Security must be applied, and roles and responsibilities as part of cloud environment must be added. Confidentiality, integrity and availability are the major principles of information security that must also be applied to the cloud, when we are dealing with business continuity [25].

[5]Cloud service provider
[6]The Open Group Architecture Framework
[7]Sherwood Applied Business Security Architecture
[8]Information Technology Infrastructure Library
[9]Committee of Sponsoring Organizations
[10]Control Objectives for Information and Related Technology
[11]Deter, Detect, Delay and Deny phases

Cloud backup and disaster recovery are very important segments of cloud storage security. The solutions for disaster recovery are based on three essentials: a scalable file system, a fully virtualized storage infrastructure and a compelling self-service recovery application [25].

### Domain 8: Data center operations

This domain is primarily focused on helping users identify common data center characteristics that could be detrimental to on-going services and fundamental characteristics for long-term stability [13].

The "Next Generation Data Centers" are improved by understanding applications in data center and requirement of hosting large scale analytical clusters. In order to manage latency, those data centers must be agile and connected to other data centers.

Organizations that build cloud data centers should implement principle cloud characteristics from Domain 1, add management processes, practices and software for technology inside the data center, and during that time improve that processes. On the other hand, cloud customer must be sure that the provider has established service management processes and practices to run his data centers, and investigate how provider's management policies and procedures will hit their environment.

### Domain 9: Incident response

The difficulty to determinate a contact person in case of security incident in the cloud, makes this domain a guideline on how to use standard incident response mechanisms with a few modifications. According to CSA, good incident handling can be reached by understanding the cloud characteristics, and addressing each of them in a specific phase of the Incident Response Lifecycle (described in the Computer security incident handling guide (NIST 800-61) [40]). Some characteristics are more challenging for incident response that the others, like on-demand self-service nature of the cloud which makes

receiving co-operations with CSP[12] very hard, the resource pooling which may complicate the forensic activities and cause privacy concerns for co-tenants and crossing the geographic and jurisdiction boundaries [25].

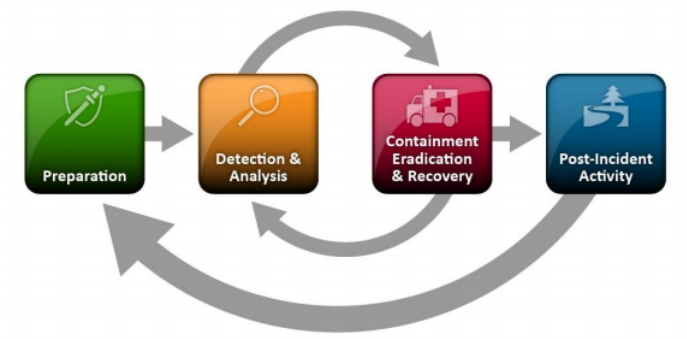


**Figure 2.4:** IR lifecycle [40]

The Incident Response Lifecycle (Figure 2.4) described in NIST 800-61 [40], gives the following phases of IR lifecycle and provides recommendation for handling with this challenges: preparation, detection and analysis, containment, eradication and recovery and post-incident activity. The preparation phase involves constitution of an incident response team and set of controls based on results of risk assessments. When a security incident is detected, the process goes into the second phase and alerts the organization and tries to mitigate the impact by containing and recovering from the incident. According to the severity of breaches, the process can rotate between those two phases. The last phase is performed when the incident is handled, a report is made with details of the cause and cost, and a recommendation for future prevention of this incident [25] [40].

### Domain 10: Application Security

This section gives instructions on how to mitigate risks and manage assurance within cloud applications, showing the security issues over the lifetime of an

[12]Cloud service provider

application – through design and operations until ultimate decommissioning. Applications can be affected by a few aspects: software development life cycle (SDLC), application security architecture, compliance, tools and services and vulnerabilities [13].

Secure SDLC has an important role in deploying an application in a cloud. Development program must contain the best practice for application security, privacy, and identity and data management. When an application is developed in cloud environment, a few aspects must be taken into consideration: control over physical security in public clouds is very low, protecting the data through the whole lifecycle must be constant, web services can cause security vulnerabilities, etc. Configuration management and ongoing provisioning need to be more complex than in traditional deployment and for that some changes in the architecture need to be done to assure more secure application. Also, it is important to support dynamic analysis of web application security tools for cloud hosted applications, and to monitor the applications more frequently. For better security, it is also recommended to update threat and trust models, to update the application assessment tools, to create different trust boundaries for IaaS, PaaS and SaaS. For IaaS, a great security success can be made with trusted virtual machine images [13] [25].

#### Domain 11: Encryption and key management

Encryption must be used to protect sensitive data. This domain focuses on identifying proper encryption usage for resource protection and scalable key management to enable access to protected resources. The encryption in cloud computing can be used in some situations [13]:

**Encrypting data in transit over networks:** There are encrypted data which transit over the Internet and have multiple identifications, like private keys, credit card numbers or passwords. This encryption can be equally applied in SaaS, PaaS and IaaS.

**Encrypting data at rest:** This is a situation when data are encrypted on

a disc or in a live production database. It is recommended for the customer to encrypt his own data before sending it to the cloud. In this example, the customer can hold and control cryptographic keys and if it's necessary decrypt the data on his own device. In IaaS environment, using some third part tools it is easy to encrypt data at rest, however, within PaaS this is more complex, and even impossible in SaaS environment if we don't have permission from the provider.

**Encrypting data on backup media:** This is a situation where data is protected against misuse from stolen or lost media. Although it would be ideal for the cloud service provider to implement that, it must be the customers' concern to do the encryption.

There are two concepts used for encryption, namely content aware and format preserving encryption. But there are also alternative approaches for organizations that send out unsecured data: tokenization, data anonymization and utilizing cloud database controls [25].

Cloud encryption must be supported by robust key management schemes. The most common issues and challenges for key management are how to secure and access the key store, and how to make key backup and recoverability [13].

### Domain 12: Identity and access management

This domain highlights managing identities and providing access control for cloud applications, with the focus on issues encountered when extending organization's identity to the cloud. Cloud-based Identity and access management (IAM) is composed of few essential functions: identity provisioning, authentication, federation and authorization and user profile management [13].

Enterprises that need user management processes must extend the security and management of on-boarding and off-boarding (provisioning/deprovisioning) of users in the cloud. It is recommended for customers to leverage standard connectors build on SPML schema and to modify their authorita-

tive repositories of identity data. Authentication of users across all types of cloud services is also a very important requirement for improving security. In this context, organizations must provide strong (multi-factor) authentication, delegated authentication and managing trust and credential. Federated identity management uses organization's chosen identity provider (IdP) to help them authenticate their users, in a way which provides single or reduced Sign-On (SSO) and secured exchanging identity attributes between the service provider and the identity provider. Organizations must address their services with regard to identity lifecycle management, authentication methods, token formats and non-repudiation. Authorization and user profile management have many aspects and can vary according to user's role – as a consumer or as a member of the organization. Adopting the access control in SPI environments needs some requirements, including establishing trusted user profile and policy information [13].

#### Domain 13: Virtualization

Cloud virtualization brings some benefits like multi-tenancy, data center consolidation and better server utilization. However, it also brings all the security concerns of the virtualized operation system, as well as new threads [13] [25].

One of the issues is proper protection of virtual machines with firewalls, antivirus, file integrity and log monitoring and web application protection. A primary concern of the hypervisor is to establish physical security and proper management of configuration and operations. Network security is another item on which virtualization has impact. Standard network security controls cannot perform monitoring or in-line blocking, because today virtual machines communicate over a hardware backplane, and that problem can be solved with in-line virtual appliance or hardware-assisted virtualization. Other issues that this domain focuses on are: performance concerns arising from CPU and memory used for security, instant-on gaps, data commingling, residential data destruction and difficulty of encrypting virtual machine images [25].

#### Domain 14: Security as a Service

Security as a service (SecaaS) uses adequate tools and expertise to provide detection, remediation and governance of security infrastructure to a trusted third party. Until now, cloud security discussions have focused on how to move to the cloud and provide confidentiality, availability and integrity. This domain points at Enterprise security. This makes SecaaS different from any other cloud security [25]. Security as a Service refers to the provision of security applications and services via cloud either to cloud-based infrastructure and software or from the cloud to the customer's on-premise systems. This will enable enterprises make security services more cost effective when in the cloud than provided locally. The following security categories covered in SecaaS are of most interest to consumers and security professionals: identity and access management (IAM), data loss prevention (DLP), web security, email security, security assessments, intrusion management, security information and event management (SIEM), encryption, business continuity and disaster recovery and network security [41].

### 2.1.4 Security issues according to SPI

According to infrastructure security in the context of SPI service delivery model, there are security issues at the network, host and application level. Infrastructure security is more applicable for IaaS customers, but it can also be applied for PaaS and SaaS, since they have ramifications to the customers' risk and compliance management [4].

#### Network level

At the network level, it is important to distinguish between public clouds and private clouds. In private clouds, there are no new attacks, vulnerabilities, or changes in risk that need to be taken into consideration. On the other hand, in public clouds, the customers' network topology must interact with cloud providers' network topology, which brings changes in security requirements.

There are a few significant risk factors in this case [4]:

- Ensuring data confidentiality and integrity.
- Ensuring proper access control (authentication, authorization, and auditing).
- Ensuring the availability of internet-facing resources.
- Replacing the established model of network zones and tiers with domains.

Data in transit, which flows to and from the public cloud provider, are exposed to the Internet, and its confidentiality and integrity must be considered. An example of this is Amazon Web Services (AWS) security vulnerability from December 2008 [42], where the use of HTTP instead of HTTPS has increased the risk of data modification without customer's knowledge.

The lack of audit of the operations of cloud provider's network decreases customer's access to relevant network-level logs and data. This risk factor may produce the issue of reused IP addresses, since the cloud providers do not "age" IP addresses enough when they are given back. In this way, the old addresses are still in the DNS caches, and allow users to reach these non-existing resources.

With the third risk factor mentioned in this section, a few security issues that need to be considered are highlighted. One of them is BGP[13] prefix hijacking (i.e., the falsification of Network Layer Reachability Information). Because of the configuration mistake, prefix hijacking is committed by announcing an autonomous system address space that belongs to someone else without their permission. As well as misconfigurations, there is also prefix hijacking due to deliberate attacks. These are rarer than misconfigurations, but can still block the access to the data. One study presented to the North American Network Operators Group (NANOG) [44] in February 2006, has

[13]Border Gateway Protocol – an interdomain routing protocol used in the core of the Internet [43]

shown that several hundred misconfigurations occurred per month, and deliberate attacks occurred on less occasions than 100 times per month. Prefix hijacking is not new, but with the increased use of cloud computing, the availability of cloud-based resources increases as well, which is the target of those malicious activities. DNS[14] attacks are not new and cloud related as well as the previous attacks, but the issue with cloud computing and DNS presents a network level security risk because of increased external DNS querying. In addition to vulnerabilities in the DNS protocol and its implementation, DNS cache poisoning attacks are big problems in the context of cloud computing. DNS server is tricked to accept incorrect information on a way that the target domain's name server (NS) is redirected, its record is redirected to another target domain and it's responded before the real NS. Another consideration must be added to denial of services (DoS) and distributed denial of services (DDoS) attacks when using IaaS. Except the external, also internal DDoS attacks can be provided through the portion of the IaaS provider's network used by the customer, and the provider would probably not have any detective controls to notify such an attack. Only customers can prevent those attacks with hardening its instances and to firewall off groups of instances [4].

Traditional network security relies on isolation model of network zones and tiers, where users and systems in specific roles have access only to specific tier or zone. However, this model cannot be applied to the public IaaS and PaaS clouds, but that approach can be replaced in public clouds with "security groups", "security domains" or "virtual data centers". Those security groups may allow virtual machines to access each other using a virtual firewall, filtering the traffic based on IP address, packet types and ports. Also, the applications are logically grouped based on domain names, for example, Google's App Engine named its services *mytestapp.test.mydomain.com* [4].

[14]Domain Name System [45]

### Host level

The host security issues are closely related to the different cloud service delivery models and deployment models. Although there are no new threats to hosts that are specific to cloud computing, some virtualization security threats are carried to the public cloud computing environment (VM escape, system configuration drift and insider threats by way of weak access control to the hypervisor). The elasticity of the cloud can bring new operational challenges from a security management perspective, which can be much harder than just running a scan [4].

***SaaS and PaaS host security***

Since hackers can use the knowledge about cloud configurations to intrude into cloud services, cloud service providers are not sharing publicly the information about their host platforms, operating systems and processes. In that way, host security is not known in detail to customers and the whole security responsibility lies on cloud service provider. Customers can ask CSP to share information under a nondisclosure agreement (NDA) or via controls assessment framework – ISO 27002 or SysTrust to give customer the assurance. To provide virtualization, the CSP engages virtualization platform with Xen and VMware hypervisor included in their host platform architecture. So that a host abstraction layer is granted, hiding the host operating system from end users. In the case of SaaS, this layer is only available to developers and CSP's operations staff, and in PaaS case, customers use PaaS API that interact with the host abstraction layer. Although the CSP is responsible for host security for SaaS and PaaS services, customer can own the risk of managing information hosted in the cloud [4].

***IaaS host security***

When using IaaS, customers have responsibilities for host security, which can be categorized as virtualization software security and virtual server security [4].

*Virtualization software security*

Managing a virtualization software is assigned to the CSP, while cus-

tomers do not have access to this software, especially if it is a public cloud service. Virtualization of hardware or operating system allows sharing of resources across multiple guest virtual machines at the same time and without interfering between each other. Host level virtualization can be accomplished using type 1 hypervisors, such as VMware ESX, Xen, Oracle VM and Microsoft's Hyper-V.

*Virtual server security*

Customers have full access to virtual instance of an operating system, which is visible to them from the Internet and isolated from other instances by hypervisor technology. For that, the security and security management of those instances is customer's responsibility. Public IaaS can be very vulnerable and cn be a victim of some new host security threads, such as stealing SSH private keys for host accessing and managing, attacking services listening on standard ports, hijacking accounts, attacking unsecured systems by host firewalls and deploying Trojans into virtual OS. For better virtual server security must use strong operational procedures coupled with automation of procedures and the following things are recommended:

- Using a secure-by-default configuration – building custom VM images that have only capabilities and services necessary to support the application stack.
- Tracking the inventory of VM image and OS versions that are prepared for cloud hosting to undertake the same level of security verification and hardening for hosts within the enterprise.
- Protecting the integrity of an image from unauthorized access.
- Guarding the host access private keys on safe place.
- Guarding the decryption keys out from the cloud where the data is hosted.
- Except for a key for decrypting the filesystem key, no other authentication credential should be included.

- Not using password-based authentication for shell access.
- Periodically reviewing logs for suspicious activities.

### Application level

Application security is a critical element of infrastructure security program. Designing and implementing applications for a cloud platform require improving of current practices and standards for existing application security programs. Involved applications vary from standalone single-user applications to sophisticated multiuser e-commerce applications used by millions of users, but the most vulnerable are web applications. Since browser has emerged as the end user client for accessing in-cloud applications and they are vulnerable to end user security attacks, it is important to include browser security into the scope of application security for achieving end-to-end security in the cloud. Hence, customers are recommended to regularly check browser updates to maintain end user security [4] [46].

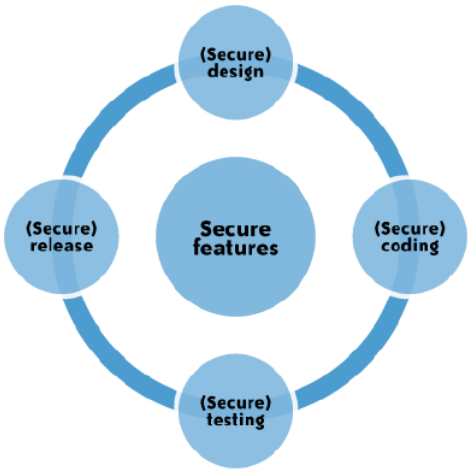


**Figure 2.5:** Software Development Life Cycle (SDLC) [4]

According to some researches from SANS [47], web applications vulnerabilities accounted for almost half of all critical security controls [48]. OWASP Top 10 from 2007 [49] to 2013 [50] shows that the main threats for application

security are SQL injection, cross-site scripting (XSS), broken authentication and session management, insecure direct object references, malicious file execution and other vulnerabilities, which are result of programming errors and design flaws. Web applications built and deployed in a public cloud platform are easy to scan with appropriate knowledge and tools by hackers, therefore those applications must be designed for an Internet threat model and security must be embedded into the Software Development Life Cycle (SDLC) drawn with secure design, coding, testing and release. Additionally, application-level DoS attacks can occur as high-volume web page reloads, XML web services requests or protocol-specific requests supported by a cloud service. The most often DoS attacks are on pay-as-you-go cloud applications and will try to increase the cloud utility bill with huge using of network bandwidth, CPU and storage. Those types of attacks are known as economic denial of sustainability (EDoS) [51].

The responsibilities about web application security in the cloud lay on both the customers and the cloud provider and depend on the cloud service delivery model (SPI) and service-level agreement (SLA). In SaaS model, the service provider is mainly responsible for his own application security, while the customer is in charge of operational security functions and user and access management. An interesting problem is the authentication and access control features offered by SaaS providers. Different providers offer different methods: web-based administration user interface tool to manage authentication and access control of the application (Salesforce.com, Google), built-in features that customers can invoke to assign read and write privileges to other users (Google Apps). Customers should accept this access control mechanisms, as well as include strong privilege management based on user roles and functions and implement a strong password policy [4].

Since PaaS clouds do not only support developing environment, but also support customer's own applications, PaaS application security can be divided into two layers: security of the PaaS platform (runtime engine) and security of customer applications deployed on a PaaS platform. Cloud service

providers are responsible for securing the platform software stack or to guaranty that the third-party application provider secures their services. They are also responsible for monitoring new bugs and vulnerabilities and monitoring of a shared network and system infrastructure of a customer's applications. PaaS developers need to understand the specific cloud providers' APIs and its security features – security objects and web services for setting up authentication and authorization tools within applications. On the other hand, it is expected from the cloud service provider to offer user authentication, single sign-on (SSO) using federation, authorization and SSl or TLS support. Since there is no PaaS security management standard, security features and models can vary between providers [4].

In IaaS clouds, customers have complete control over securing their applications, because the entire stack runs on customer's virtual servers, and from cloud provider they should receive only basic guidance referring to firewall policy. Web applications deployed into the public IaaS cloud must be designed for an Internet thread model, and should be periodically tested for vulnerabilities and security must be embedded into the SDLC. IaaS developers need to implement their own features to handle authentication and authorization and should make their application supportive for authentication service features by an enterprise Identity Provider or third-party identity service provider.

## 2.2 Data security

### 2.2.1 Data life cycle

The idea of protecting data in the cloud is similar to traditional data security, but because of openness and multitenant characteristic of the cloud, the content of cloud data security has its particularities. The problem of keeping data secure and confidential is shown through the Data Security Lifecycle [20] [8].

Data lifecycle refers to the entire process from generation to destruction


**Figure 2.6:** Data Security Lifecycle [39]

of the data. The data life cycle is divided into six stages [39] [25]:

**Create**

Creation is the generation of new or modifying the existing digital data element, so can be named also as Create/Update phase. It can be every kind of content, not just a document or database, and can be either structured or unstructured. In this phase the information is classified and the appropriate rights are determined.

**Store**

Storing is the act committing the digital data to structured or unstructured sort of storage repository (database vs. files) and usually occurs simultaneously with creation. Here the classification and rights to security controls are mapped, including access controls, encryption and rights management.

**Use**

Data is viewed, processed, or otherwise used in some sort of activity, not including modification. These controls apply to data at the point of use- typically a user's PC or an application. There are detective controls like activity monitoring, preventative controls like rights man-

agement and logical controls which are typically applied in databases and applications.

**Share**

Data is made accessible to others and is exchanged between users, customers, and partners. These controls include a mix of detective and preventative controls, encryption for secure exchange of data, and logical controls and application security.

**Archive**

Data leaves active use and enters long-term storage. Here, the protection of data and its availability is ensured with a combination of encryption and asset management.

**Destroy**

Data is permanently destroyed using physical or digital means (e.g., crypto-shredding). The data must be deleted securely and need to be used tools to track down any lingering copies.

### Locations and access

One significant gap in the data lifecycle is that it doesn't adequately show the location of the data when is moving between repositories, environments and organizations and how it is accessed during the phases.

The role of the location can be represented by thinking of the lifecycle as a series of smaller lifecycles running in different operating environments and not as a single, linear operation. Data can be moved into, out of, and between these environments at any phase of data lifecycle. Strong data security can be achieved by identifying these movements and applying the right controls at the right security boundaries. These locations may be internal, external, public or private cloud, cloud provider or traditional outsourcers etc., and for that it is extremely important to understand both the logical and physical locations of data, due to the potential regulatory, contractual and other jurisdictional issues.

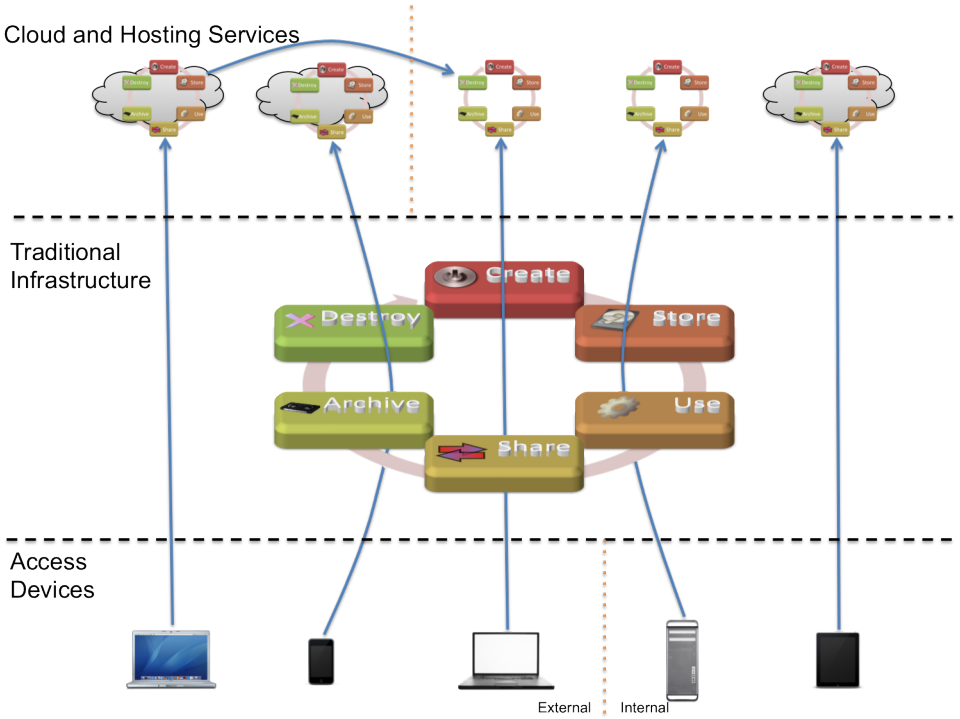


**Figure 2.7:** The role of the location into the Data Security Lifecycle [39]

For fully completing the data lifecycle, despite location it is needed to be known who have access to the data and how can they access it (device and channel). Data can be accessed from different devices which have different security characteristics and may use different applications or clients [39].

### Functions, actors and controls

The last step for completing data lifecycle is to add functions that can be performed with the data, by a given actor and on a particular location. An actor can be a person, application or system, as opposed to an access device. The actor can perform three functions [39]:

**Access:** View/access the data, including copying, file transfers, and other exchanges of information.

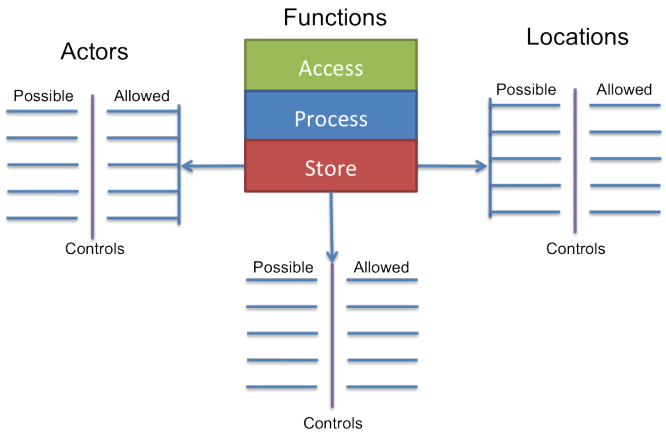


**Figure 2.8:** Completing the data lifecycle with functions, actors and controls [39]

**Process:** Perform a transaction on the data: update it, use it in a business processing transaction, etc.

**Store:** Hold the data (in a file, database, etc.).

The table below shows which functions map to which phases of the lifecycle:

| | Create | Store | Use | Share | Archive | Destroy |
|---|---|---|---|---|---|---|
| Access | X | X | X | X | X | X |
| Process | X | | X | | | |
| Store | | X | | | X | |

**Figure 2.9:** Mapping the functions into the different phases of the lifecycle [39]

A control restricts a list of possible actions down to allowed actions. For example, encryption can be used to restrict access to data, application con-

trols to restrict processing via authorization, and DRM storage to prevent unauthorized accesses [39].

### 2.2.2 Aspects of data security

Ensuring data safety in the cloud requires different approaches and good understanding of different stages of data transmission [4]:

- data in transit,
- data at rest,
- processing of data, including multitenancy,
- data lineage,
- data provenance,
- data remanence.

The first three stages can be compared trivially with example of using an email, where sending an email represents data in transit, data at rest is the email in the mailbox and processing of data refers to typing up a response [52].

It is recommended to protect the data in transit using vetted encryption algorithm, especially when using a public cloud. Using encrypted data with a non-secured protocol can provide confidentiality, but does not ensure the integrity of the data. Hence, it is essential to use a protocol that will provide integrity as well, such as FTP over SSL (FTPS), Hypertext Transfer Protocol Secure (HTTPS) or Secure Copy Program (SCP) [4].

Encrypting data at rest is not as simple as data in transit. Encryption is possible when using an IaaS cloud service for simple storage, but using encryption in PaaS and SaaS cloud application is not always feasible, because it prevents indexing and searching of that data. However, when applications work with data processing, this data must be unencrypted, and that

is a scenario where the organization's data are definitely not encrypted. For now, a fully homomorphic encryption scheme is being developed [53] which allows data to be processed without being decrypted, although some earlier researches shows that homomorphic encryption required immense computational effort [54].

When data are in the cloud, it is recommended to follow its history, to know exactly where and when the data were located within the cloud. A data lineage analysis can be performed with ETL[15] tools and can track data alterations by presenting a series of data output-input dependencies within such an environment as a graph of nodes and links [55]. This data path visualization is known as data lineage. Although providing data lineage is very important for auditor's assurance, this process is time-consuming and for a public cloud it is not really possible [4].

A more challenging problem for the customers is providing data provenance, which means not just proving the integrity of the data, but more specific provenance of the data. Data provenance represents a sort of metadata, containing the derivation history of a data product starting from its original sources in a data warehouse, track the creation of intellectual property and provide an audit trail for regulatory purposes [56].

A final aspect of data security is data remanence – the residual representation of digital data that remains even after attempts to remove or erase the data. Some operating systems do not delete the data immediately when the user requests that, but move it to a holding area for easily recover from a possible crash or mistake. This residue may be due to data being left intact by a nominal delete operation or through physical properties of the storage medium that allow previously written data to be recovered. Data remanence may make inadvertent disclosure of sensitive information possible, if a storage media would be released into an uncontrolled environment (e.g., thrown in the trash, or lost) [57].

[15] Extract, Transform and Load

Cloud computing with its virtualization characteristic complicates data remanence. Overwriting the physical media is virtually impossible, since the cloud infrastructure may distribute customer's storage or virtual machine instance across multiple physical drives. Furthermore, the data written on particular drives remain until cloud provider reallocates those sectors to other instances, which allow hackers to read this data by looking at the bits in the new instance. For this problem, the first recommendation is to encrypt data before it is stored within the cloud. For the keys, it is recommended to be managed locally. Using these recommendations, data can be securely deleted simply by deleting the key. This approach is sufficient if cloud is used only for storing the encrypted data, not to process it, since the data need to be decrypted [58].

From the aforementioned, it can be said that the only workable option for data security mitigating is to ensure that the sensitive data won't be stored into a public cloud, or if it must, to be used encryption with locally keeping keys.

# Chapter 3

# Cryptography deployment in clouds

## 3.1 Cryptographic cloud storage

An important aspect of a cryptographic storage service is that the security properties like confidentiality, integrity, availability, are achieved based on strong cryptographic guarantees, which is different than in legal, physical and access control mechanisms. Cloud storage can be categorized into two classes: cloud storages that are designed using cryptographic techniques but not in the framework of cryptography theory (Class-A), and cloud storages that are designed using cryptographic techniques and also in the framework of cryptography theory (Class-B). Class-B schemes were proven secure in the framework of provable-security theory in cryptography field and an example of these schemes are Kamara et al.'s and Chow et al.'s scheme. A few Class-A schemes are known today: Kamara et al.'s scheme, Barua et al.'s scheme, Kumbhare et al.'s scheme, Zarandioon et al.'s work, Somorovsky et al.'s scheme etc. [59].

### 3.1.1 Kamara et al.'s scheme

The best known storage scheme is Kamara et al.'s scheme [60]. Kamara shows a possible architecture for a cryptographic storage service and describes designs for both consumer and enterprise scenarios by using non-standard cryptographic techniques, such as attributed encryption, searchable encryption, etc. The architecture contains three components: data processor (DP) that processes data before it is sent to the cloud, data verifier (DV), that checks whether the data in the cloud has been tampered with, and a token generator (TG) that generates tokens which enable the cloud storage provider to retrieve segments of customer data [60].

#### Architecture for consumer scenario

Cryptographic cloud storage in consumer scenario includes a user Alice, the owner of the data in the cloud, a user Bob, and a cloud storage provider. In this scenario Alice wants to share data with Bob (Figure 3.1).

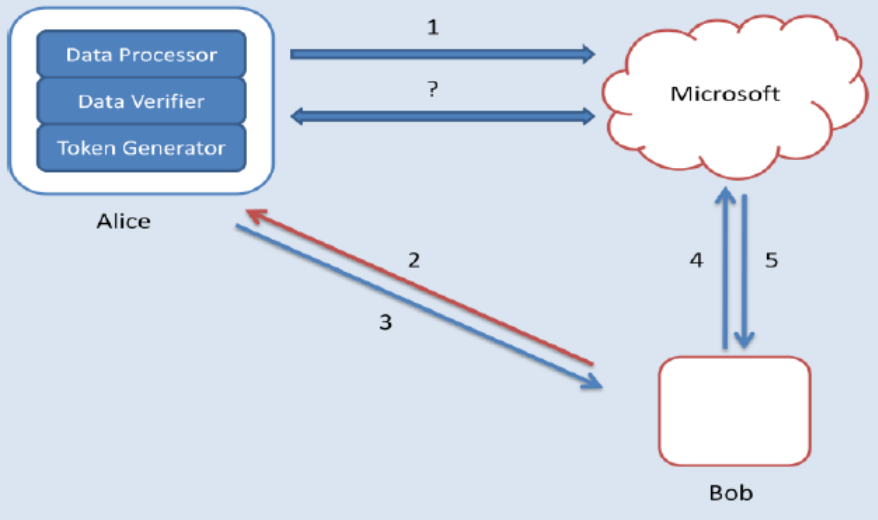


**Figure 3.1:** Architecture for consumer scenario [60]

First, Alice's data processor prepares the data before sending it to the cloud, generates cryptographic key (master key) which is expected to be

stored locally, attaches some metadata, encrypts the data with a variety of cryptographic primitives, and encodes the encrypted data and index. To share the data with Bob, Alice's token generator creates a token for the keyword and a decryption key, which are sent to Bob. Then he sends the token to the cloud, and the provider uses it to retrieve the requested encrypted files and returns them to Bob. After receiving the file from the cloud, Bob can use decryption key to decrypt the files and use them. No matter what phase the process is in, data integrity can be always verified by Alice's data verifier [60].

### Architecture for enterprise scenario

Cryptographic cloud storage in enterprise scenario includes an enterprise MegaCorp that is owner of the data in the cloud, a business partner PartnerCorp, and a cloud storage provider. This architecture is extended with new component, a credential generator (CG), who implements an access control policy by issuing credentials to parties inside and outside MegaCorp (Figure 3.2).

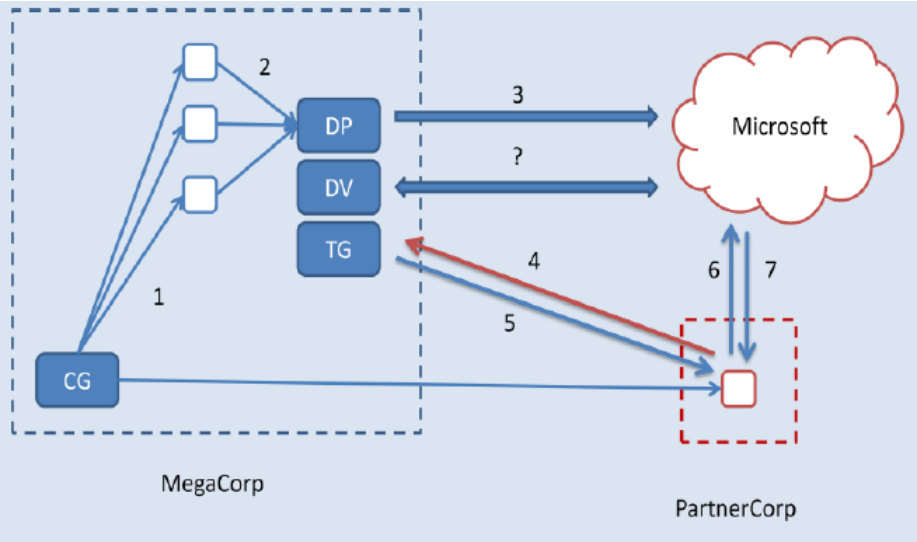


**Figure 3.2:** Architecture for enterprise scenario [60]

Each MegaCorp and PartnerCorp employee receives a credential from the credential generator, which can be their organization, team or role, and send the data to the dedicated machine along with an associated decryption policy (only members of a particular team can decrypt the data). As in the consumer scenario, the data processor prepares the data before sending it to the cloud. The PartnerCorp employee authenticates itself and sends a keyword to a MegaCorp's dedicated machine when he needs access to the data. Than the employee is checked if he has the appropriate role to access the data, and if this succeeds, the dedicated machine returns a token. This token is used by cloud to find the encrypted documents and returns them to the employee. After that, the employee uses his credentials to decrypt the file. At any point in time, MegaCorp's data verifier can verify the integrity of MegaCorp's data [60].

### Implementation and benefits

Some of the cryptographic techniques used in this architecture are developed specifically for cloud computing. The data processor uses symmetric encryption scheme (e.g. AES) under unique key for data indexing and encrypting. Next, the data processor encrypts the index using a searchable encryption scheme and the unique key using attribute-based encryption scheme. Finally, the whole data is encoded in a way that the data verifier can verify the integrity using a proof of storage.

The main benefits of using this cryptographic storage service are that the control of the data is maintained by the customer and the security properties are derived from cryptography, as opposed to legal mechanisms, physical security or access control. Moreover, this approach satisfies the main concerns from the Cloud Security Alliance's reports, such as: regulatory compliance, geographic restrictions, subpoenas, security breaches, electronic discovery, data retention and destruction [60].

### 3.1.2 Barua et al.'s scheme

Considering the problem of patient self-controlled access privilege to highly sensitive Personal Health Information (PHI), Barua proposed access control scheme for cloud storage, known as efficient and secure patient-centric access control (ESPAC) scheme, based on ciphertext-policy attributed-based encryption and identity-based encryption. This scheme allows users to have different access privileges according to its roles, and then assigns different attributes sets to them [59].

This model contains some entities: trusted authority (TA) generates the public and secret key parameters for the ESPAC and grants differential access rights to individual users based on their attributes and roles, cloud service provider, registered user – patient, who is responsible for encrypting the sensitive PHI and data-access requester – cloud user, who wants to access some specific PHI [61].

#### ESPAC scheme

In the ESPAC scheme, there are two major phases as shown in Figure 3.3. In phase A, a secure data communication between different eHealt users is established with identity-based encryption scheme. In phase B, an attribute based encryption scheme is employed to realize control of data requester's access.

In the phase-A, the communication between a remote user and an eHealth service provider follows the next steps [61]:

**Step 1 (System initialization):** The cloud service provider receives a unique ID by a trusted authority, computes the public key, generates the hash function and computes the key for message encryption and decryption. After that it generates secure hash functions and remote user's pseudo-identity and sends them to its subscribers. Trusted authority computes public and master key for attribute-based encryption and decryption. The individual user computes the public key, the session key and the

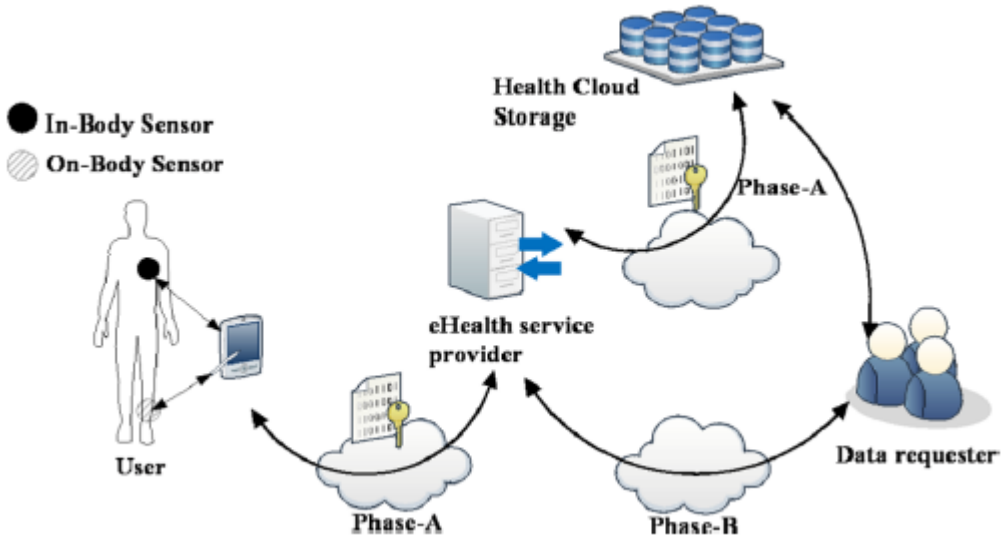


**Figure 3.3:** ESPAC scheme [61]

message token and along with the encrypted data sends them to the receiver.

**Step 2 (Secure message communication):** After the system initialization, both parties use data encryption and decryption algorithms to securely transmit their data.

**Step 3 (Message signature and verification):** After receiving the message, the receiver will verify it to ensure data integrity with cryptographic digital signature. By doing so, the eHealth service provider can verify the data originated from the specific patient and cannot be altered after signing it.

In the phase-B, an access tree is created, based on different roles of data requesters. The requester can decrypt the PHI data only with a secret key, which can be obtained if they provide the corresponding attributes - nodes of the access tree. The access tree classifies the data requesters as health workers, physicians, researchers, insurance companies, etc., and each of them

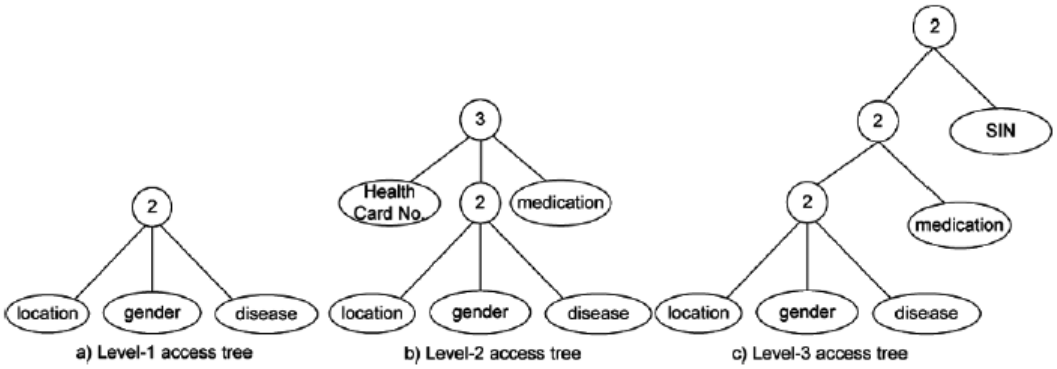


**Figure 3.4:** Access structures based on different privacy levels [61]

has a specific area of interest. Figure 3.4 shows possible access structures based on different privacy levels, where intermediate nodes work as logic gates. For example, "2 of (location, gender, disease)" in Figure 3.4(a) can be converted to "(location AND gender) OR (gender AND disease) OR (disease AND location)" [61].

From a security aspect, it can be said that ESPAC scheme satisfies some main security concerns: the ESPAC scheme ensures user and eHealth agent's identity privacy, is secure against chosen ciphertext-only attack, is resistant to eavesdropping and collusion attacks, the scheme ensures message integrity, non-repudiation, and source authentication, ensures backward and forward secrecy, etc [61].

### 3.1.3 Kumbhare et al. scheme

Kumbhare presents the Cryptonite architecture for a secure data repository service design on top of a public cloud infrastructure. The scheme allows secure storing and sharing of the cloud data, without revealing the plain text to unauthorized users. The system masks file names, user permissions and access patterns while providing auditing capabilities with provable data updates [59].

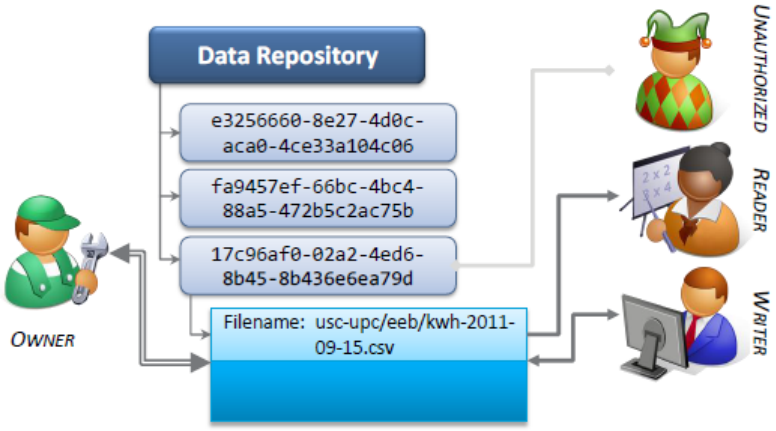


**Figure 3.5:** Entities and roles in Kumbhare's scheme [62]

In this model, a few entities and roles are defined(Figure 3.5): user, which can be categorized as "owner", authorized user ("Reader" or "Writer") or unauthorized user, cloud storage service provider for storing the files, and secure data repository (Cryptonite), which uses the cloud storage service as the backend store and manages user accounts and enforces access rights [62].

### Cryptonite architecture

The Cryptonite Architecture integrates client-side libraries with the secure data repository service and existing cloud storage service to support the operations: PUT, GET, DELETE, GRANT, REVOKE, and SEARCH (Figure 3.6).

The Cryptonite Client Library (CCL) has in authority to encrypt and preprocess the plain text files before uploading them to the Cryptonite repository service, and to decrypt it upon receipt. The client library runs on the user's local machine and forms the secure operations provided by Cryptonite. The Cryptonite Data Repository Service runs within a Cloud VM and has three major subcomponents: the File Manager (FM), the Secure Index Manager (SIM) and the Audit Manager (AM). The File Manager has responsibility to

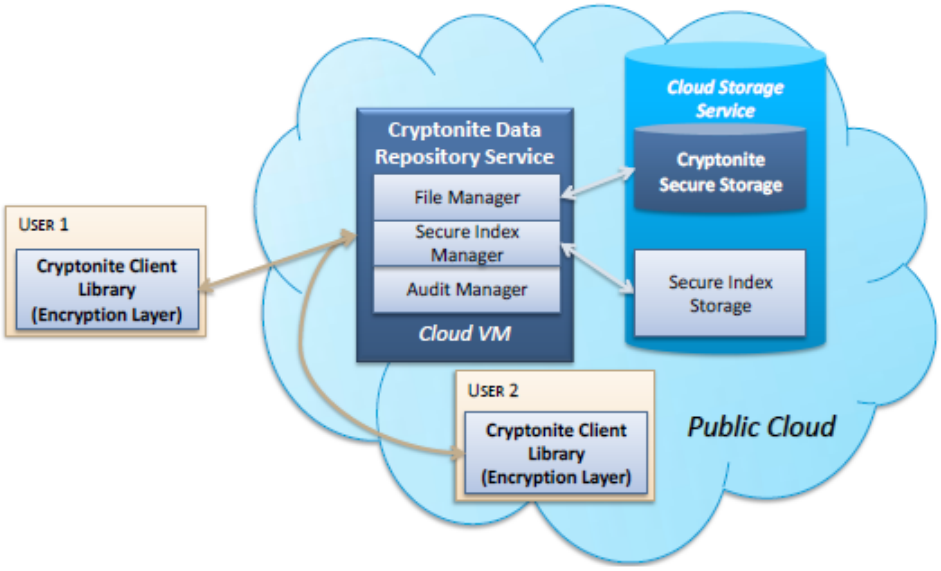


**Figure 3.6:** Cryptonite Architecture [62]

interact with the Cloud storage service to store and retrieve files requested by the user, or to restrict unauthorized updates to the stored data. The Audit Manager maintains a secure audit log for each file access performed by the File Manager, including a signature of the requested operation's message. The Secure Index Manager (SIM) keeps a secure index per user for all the files that are owned by that user, and stores this index in a separate Secure Index Storage (SIS) space. The Cryptonite Secure Storage (CSS) is a storage account within the public cloud used to store the files and metadata. The CSS uses standard authentication mechanisms provided by cloud data services to access this space and use it to store at collection of binary files [62].

### Implementation and benefits

Cryptonite uses several cryptographic and security techniques in its secure design: public key infrastructure (PKI), digital signature, broadcast encryp-

tion, lazy revocation and searchable encryption. Public key infrastructure is used to provide User Identity to each user and to allow the user to associate the public/private key pair with that identity. Digital signature is deployed in file manager and audit manager for the purposes of integrity verification as well as auditing purpose. Broadcast encryption is deployed in Cryptonite client library and allows users to encrypt their data in a way that it can be decrypted only by a particular subset of users. Lazy revocation is a strategy in which a file, that has read access rights for a user removed by its owner, is not re-encrypted unless the file's contents change. Searchable encryption is deployed in the secure index manager to allow searching within an encrypted file without decrypting the entire file [62].

This cloud architecture scenario addressed a few security issues: data storage security, metadata storage security, owner control of data sharing, data integrity and audit, masking access control list, masking access patterns, assured file deletion, etc [59].

### 3.1.4 Other cloud storage schemes

#### Zarandioon et al.'s work

Zarandioon proposed a user-centric privacy-preserving cryptographic access control protocol called key to cloud (K2C) [63], using attributed-based encryption and signature. This protocol allowing end-users anonymously and securely to store, share, and manage their sensitive data in untrusted cloud storage (Figure 3.7).

In meta-data directory, all hierarchies and data objects have access revision for reading and writing. Attribute-based cryptography is applied right in these components to realize access control [59].

#### Somorovsky et al.'s scheme

Somorovsky proposed a secure solution Sec2 [64] for distributed public cloud storage by using extensive makeup language (XML) encryption, which system

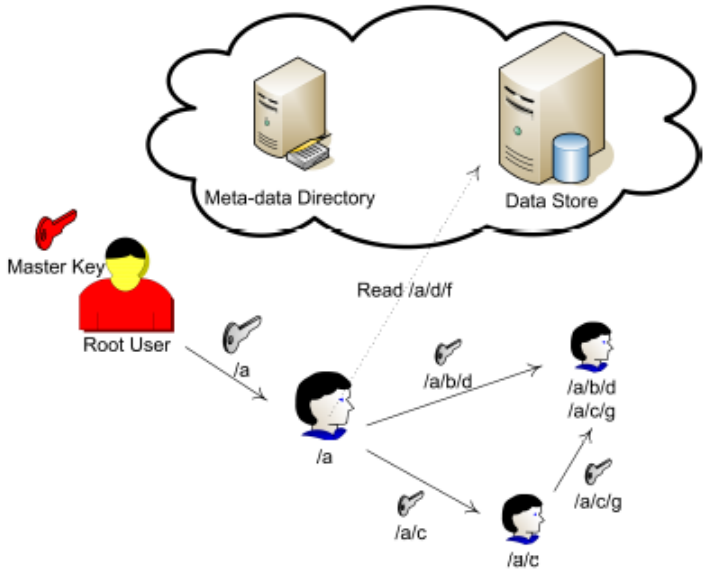


**Figure 3.7:** K2C Architecture [63]

architecture is showed in the Figure 3.8. XML encryption is used in XML encryption engine which is responsible for encryption and decryption of XML payload and key data [59].

### Popa et al. scheme

Popa [65] proposes a secure cloud storage system – CloudProof, trying to fix lack of security guarantees in service level agreements (SLAs) of cloud storage systems. This system can allow users not only to detect violations of integrity, write-serializability, and freshness, but also to prove the occurrence of these violations to a third party. This cloud storage can be implemented based on conventional cloud storage services. Furthermore, the system address broadcast encryption and key rolling to achieve key distribution in access control of CloudProof [59].

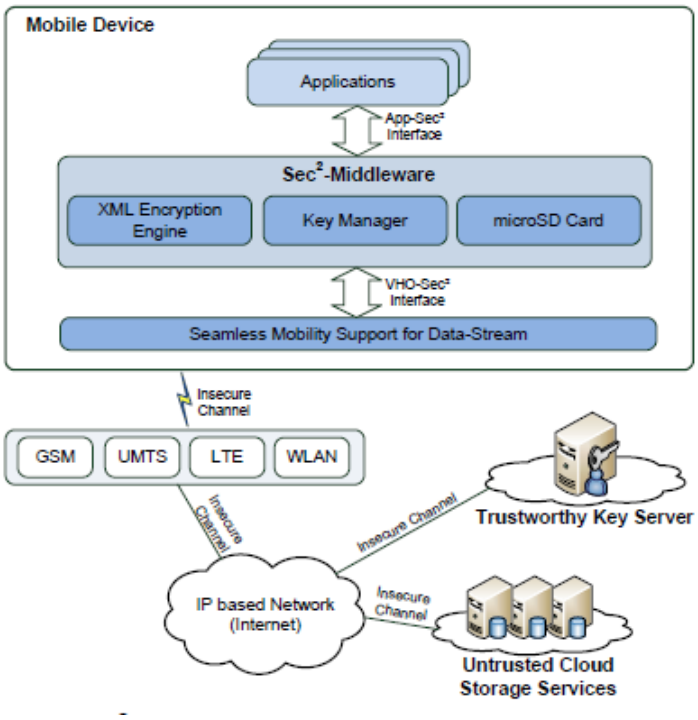


**Figure 3.8:** Sec2 Architecture [64]

### Ruj et al.'s work

Ruj's work [66] addresses the design of access control in cloud storage. A distributed access control (DACC) is ensured by employing attribute-based encryption. In DACC, cloud knows the access structure used by the owner and the attributes of the users, and the revocation of users is allowed without redistributing decryption keys to all the users in the network [59].

### Group encryption

Group encryption [67] is a useful cryptographic primitive in the scenario whenever a recipient within a group of legitimate receivers needs to be concealed. To correct the weakness that some vulnerabilities can potentially lead to disputation in cloud storage services, Feng [68] proposed a multi-

party non-repudiation (MPNR) protocol by using group encryption that is specifically designed for the cloud storage environment. This protocol focuses on how to ensure integrity with fair non-repudiation. Later, based on the same cryptographic technique, Feng et al. [69] further proposed a fair multi-party non-repudiation (MPNR) using group encryption to solve the problems of fair non-repudiation and roll-back attacks [59].

**Kamara et al.'s scheme (Cloud storage of Class-B)**

By using symmetric searchable encryption (SSE), search authenticators and proofs of storage (PoS), Kamara [70] proposed a cloud storage system CS2 from a cryptographic standpoint. CS2 can ensure confidentiality, integrity, and verifiability without sacrificing utility. In this system, SSE is used to encrypt data by a client so that it can later generate search tokens for a storage provider, and the search authenticator is used by the cloud provider to prove to the client that it returned the correct files only [59].

**Chow et al.'s scheme (Cloud storage of Class-B)**

Chow [71] proposed secure cloud storage with properties of data provenance and support of dynamic users from the cryptographic standpoint. This security model is based on cryptography theory which includes confidentiality, anonymity, and traceability. The security of Chow et al.'s scheme can be proven under some number-theoretical assumptions in the framework of provable-security theory in cryptography field. Moreover, Chow et al.'s scheme is pairing-based cryptographic cloud storage and designed using VLR (verifier-local revocation) group signature scheme and a variant of identity-based broadcast encryption [59].

## 3.2 Cryptographic techniques for cloud computing

### 3.2.1 Searchable Encryption

This encryption scheme provides encrypting a search index generated over a collection of files, in a way that its contents are hidden except to a party that has appropriate tokens. Using a searchable encryption scheme, the index is encrypted in such a way that the pointers to the encrypted files that contain the keyword can be retrieved by a given token for a keyword and without a token the contents of the index are hidden. In addition, the tokens can only be generated with knowledge of a secret key. The retrieval procedure reveals nothing about the files or the keywords except that the files contain a keyword in common. There are many types of searchable encryption schemes: symmetric, asymmetric, efficient and multi-user symmetric searchable encryption [60].

#### Symmetric searchable encryption

SSE (introduced in Song, Wagner and Perrig 2000 [72]) is appropriate in any setting where the party that searches over the data is also the one who generates it, referring to such scenarios as single writer/single reader (SWSR).

SSE schemes are based on symmetric primitives like block ciphers and pseudo-random functions and typical usage scenarios allow the data to be pre-processed and stored in efficient data structures. A SSE scheme is a set of four polynomial-time algorithms: Keygen, BuildIndex, Trapdoor, Search. Keygen($1^k$) is a probabilistic key generation algorithm, which is run by a user to setup the scheme and takes a security parameter k and returns a secret key K. BuildIndex (K,D) is an algorithm run by a user to generate indexes and as inputs it takes a secret key K and a document collection D. Trapdoor(K,w) is run by a user to generate a trapdoor for a given word and takes a secret key K and a word w as inputs and returns a trapdoor Tw. Search(I, Tw) is

run by a server S in order to find the word w into the document collection D [73].

The main advantages of SSE are efficiency and security while the main disadvantage is functionality. SSE schemes are efficient both for the party doing the encryption and for the party performing the search. The main disadvantage of SSE is known solutions trade off efficiency and functionality. The search time for the server is optimal, but updates to the index are inefficient [73]. On the other hand, in the scheme proposed by Goh [74], updates to the index can be done efficiently but search time for the server is slow [60].

### Asymmetric searchable encryption

ASE schemes (introduced in Boneh 2004 [75] and improved in Abdalla 2005 [76], Park 2005 [77] and Boneh 2007 [78]) are appropriate in any setting where the party searching over the data is different from the party that generates it, referring to such scenarios as many writer/single reader (MWSR).

ASE schemes consist of the following polynomial time randomized algorithms: KeyGen(s), run by the user, takes a security parameter s, and create public/private key pair $A_{pub}$, $A_{priv}$; PEKS ($A_{pub}$;W)[1] which produces a searchable encryption of a word W; Trapdoor($A_{priv}$;W) to produce a trapdoor $T_W$ for any keywords W that user wants the server to search for, and Test($A_{pub}$; S; $T_W$) algorithm to determine whether a given document contains one of the keywords W specified by the user [75].

The main advantage of ASE is functionality - ASE schemes are usable in a larger number of settings than SSE schemes. The main disadvantages are inefficiency and weaker security, because all known ASE schemes require the evaluation of pairings on elliptic curves which is a relatively slow operation compared to evaluations of cryptographic hash functions or block ciphers [60].

[1] Public Key Encryption with keyword Search

#### Efficient ASE

ESE schemes [79] are appropriate in any setting where the party that searches over the data is different from the party that generates it and where the keywords are hard to guess (MWSR scenario). The main advantage of ESE is that search is more efficient than ASE, and the main disadvantage is that ESE schemes are also vulnerable to dictionary attacks [60].

#### Multi-user SSE (mSSE)

mSSE schemes [73] are appropriate in any setting where many parties wish to search over data that is generated by a single party, referring to such scenarios as single writer/many reader (SWMR). The owner of the data also can add and revoke users' search privileges over his data [60].

### 3.2.2 Attribute-based Encryption

Attribute-based encryption was introduced in Sahai and Waters 2005 [80] and is also known as fuzzy identity-based encryption, where an identity is a set of descriptive attributes (e.g., roles, age, relationship, trust, location, etc.).

This encryption is a set of cryptographic techniques that allow the specification of a decryption policy to be associated with a cipher text – each user in the system is provided with a decryption key that has a set of attributes associated with it. A user can then encrypt a message under a public key and a policy. ABE uses a tree-based access structure, which allows the encryptor to specify which attributes can decrypt the data. Decryption will be possible only if the attributes associated with the decryption key match the policy used for encryption. The tree-based access structure uses operators such as AND, OR and k-of-n, where AND is usually known as n of n and OR is known as 1 of n [81].

ABE was improved with two types of variant of ABE scheme: key-policy attribute based encryption (KP-ABE) [82] and cipher-policy attribute based

encryption (CP-ABE) [83].

### 3.2.3 Proofs of Storage

A proof of storage is a protocol executed between a client and a server, which allows a client to verify that a server faithfully stores a file. Proofs of storage can be constructed from any homomorphic linear authenticator (HLA). Proofs of storage were introduced in Ateniese 2007 [84] and Juels 2007 [85] and improved in Shacham 2008 [86], Ateniese 2009 [87] and Erway 2009 [88].

Before storing the data on the cloud, the client encodes it, and after that, whenever it wants to verify the integrity of the data it runs a proof of storage protocol with the server. The main benefits of a proof of storage are that they can be executed an arbitrary number of times and the amount of information exchanged between the client and the server is extremely small and independent of the size of the data.

Proofs of storage can be either privately or publicly verifiable. Privately verifiable proofs of storage only allow the client to verify the integrity of the data. With a publicly verifiable proof of storage, on the other hand, anyone that possesses the client's public key can verify the data's integrity [60].

# Chapter 4

# DropBoxCrypt Application

## 4.1 Application purpose and specification

Cloud computing has been expanding its influence in mobile computing practices. Thousands of mobile applications integrate various cloud storage providers into their software, allowing users to store data in the cloud. However, when a security issue is in question, the cryptography must be taken into consideration. While there are countless data encryption and decryption tools available across most popular computer platforms, choices are rather limited for mobile users. Android is no exception in this regard. There are some applications that have integrated the encryption and decryption functionality on Android devices, such as Cryptonite and Boxcryptor. Those applications add an extra level of protection, but from a security and performance perspective, they need to be improved. The purpose of this chapter is to show how we can protect our cloud data through our application via our mobile devices, and to propose a new approach to user authentication, enabled by new technical developments.

DropBoxCrypt application represents a data encryption tool for mobile devices that can be used to browse files, to encrypt the chosen files and to upload them into the Dropbox storage and to decrypt them when they are downloaded. Thus, users can store data in the cloud in a secure manner.

Our application is developed in Android Studio v.1.3.2 which is compatible with Android devices using API 23. It connects with our Dropbox storage using its Dropbox API. Dropbox uses OAuth 2, an open specification for user's authentication and making secure communication between the mobile application and the cloud storage. Before decrypting the file, we authenticate the user with their fingerprint, a feature that is allowed in the last Android version – Android 6 Marshmallow, where the Fingerprint API is introduced. The code is available on `https://github.com/ikostadinovska/dropboxcrypt`.

## 4.2 Use – case scenarios

The first step in using this application is to make a user authentication. Within the main window, we connect the application with our Dropbox account. There are two use-case scenarios: upload and download file.

### 4.2.1 Upload file

If we want to upload a file, we choose the Upload button. After that, we search through the file system and choose a file that we want to upload into the cloud. When this is done, the application makes encryption to that file and sends it like that to the Dropbox. At the same time, a .key file is made for future decryption and stored on the device storage.

### 4.2.2 Download file

If we want to download a file from the Dropbox storage, we choose the Download button. At this stage, the application shows another view for fingerprint authentication and prompts us to touch the fingerprint sensor. Then the application starts listening for a fingerprint. When the fingerprint is available, the application checks if a fingerprint or a system password exists. The fingerprint dialogue also allows the use of a backup password, if it is our

choice, or in the case of a hardware error. After successful authentication, the user is redirected to the Dropbox storage. From the /Apps/DropBoxCrypt/ folder we choose the wanted file. The file is downloaded to the phone storage, the appropriate .key file is found and the decryption is done. Before using this application, the fingerprint must be registered on the device with its backup password.

## 4.3 Application development

### 4.3.1 Application structure and architecture

The application consists of several activities (Figure 4.1):

**MainActivity** links the whole application together and gives the application its basic view. Here the connection with Dropbox is done, using OAuth protocol with OAuth2 and OAuth1 access token. In the onActivityResult() method, in case that we choose a file from dropbox, we authenticate with our fingerprint in the onRequestPermissionsResult() method, which handles the permissions. After permission is granted, we make a DownloadFile object, which is used to download the selected file. In case we choose a file from the phone storage, we make DataEncryptionCrypto object to encrypt the chosen file, and UploadFile object to send that file to the dropbox.

**DownloadFile** is a class for downloading the selected file from the Dropbox storage. When we reach the file, we create DataEncryptionCrypto object to decrypt the file.

**UploadFile** is a class for uploading a file into the Dropbox (encryption is already made before in MainActivity).

**DataEncryptionCrypto** contains encryptFile() and decryptFile() methods. For its purpose we use javax.crypto with its Chiper and SecretKey classes with AES 128.

**KeyStoreUtils** is used for managing the keys, and is compose of generateKey(), saveKey() and loadKey() methods.

**FingerprintAuthenticationDialogFragment** handles the dialog which uses fingerprint APIs to authenticate the user.

**FingerprintUiHelper** is a small helper class that manages text and icon for fingerprint authentication UI.

**FingerprintModule** is a dagger module for Fingerprint APIs.

**InjectedApplication** is a class of the sample that holds the ObjectGraph in Dagger and enables dependency injection.

The application also has two subprojects:

**aFileChooser** chooses files from the phone storage.

**dropboxChooserSDK** searches and selects files into the Dropbox storage. The Chooser [89] is the fastest way to get files from Dropbox into third-part application. It's a small JavaScript component that enables our application to get files from Dropbox without having to worry about the complexities of implementing a file browser, authentication, or managing uploads and storage.

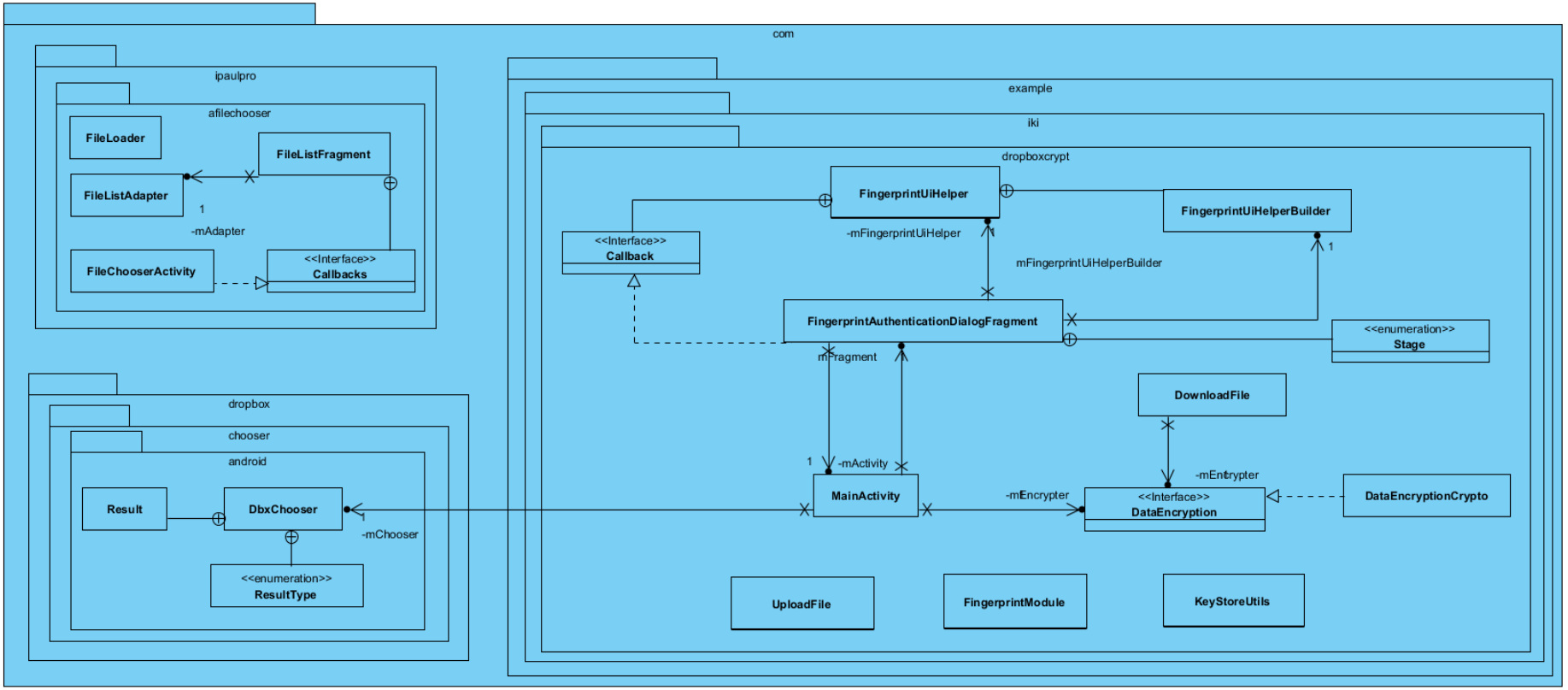


**Figure 4.1:** UML Class Diagram for DropBoxCrypt application

### 4.3.2 Application Flowchart

The following steps illustrate how DropBoxCrypt application works (Figure 4.2):

1. User enters the application.
    - I Application connects to Dropbox API.
        - i. Application checks if the right Dropbox application key is used.
    - II Application waits for user's commands.
2. User attempts to log in into Dropbox.
    - I Application firstly checks if the user has been logged out from previous session.
    - II Application requires user's credentials for login process.
3. User enters username and password.
    - I Application checks user's credentials.
        - i. Remote OAuth2 authentication starts.
    - II User logs in.
    - III Application waits for user's further commands.
4. User attempts to upload file to Dropbox.
    - I Upload process starts.
    - II When upload process is done, application returns back to home screen.
5. User attempts to download file from Dropbox.
    - I Download process starts.

II When downloading is done, application returns back to home screen.

6. User attempts to log out.

I Application checks if the user is still logged in.

i. Application removes user's credentials from the current session.

II User logs out.

III Application returns back to home screen.

The following steps illustrate the Upload process (Figure 4.3):

1. Application shows a file chooser.

I User selects a file for the upload process.

II Application uses file path of the targeted file.

2. Application encrypts the selected file.

I Application sets the block size.

i. Application creates file's key.

ii. Application creates cryptographic cipher for encryption and decryption using AES.

II Application uses encrypted file's file path.

3. Application uploads the encrypted file to Dropbox's storage.

The following steps illustrate the Download process (Figure 4.4, Figure 4.5 and Figure 4.6):

1. During the initialization, Android device checks application's permissions stored in application's Manifest file.

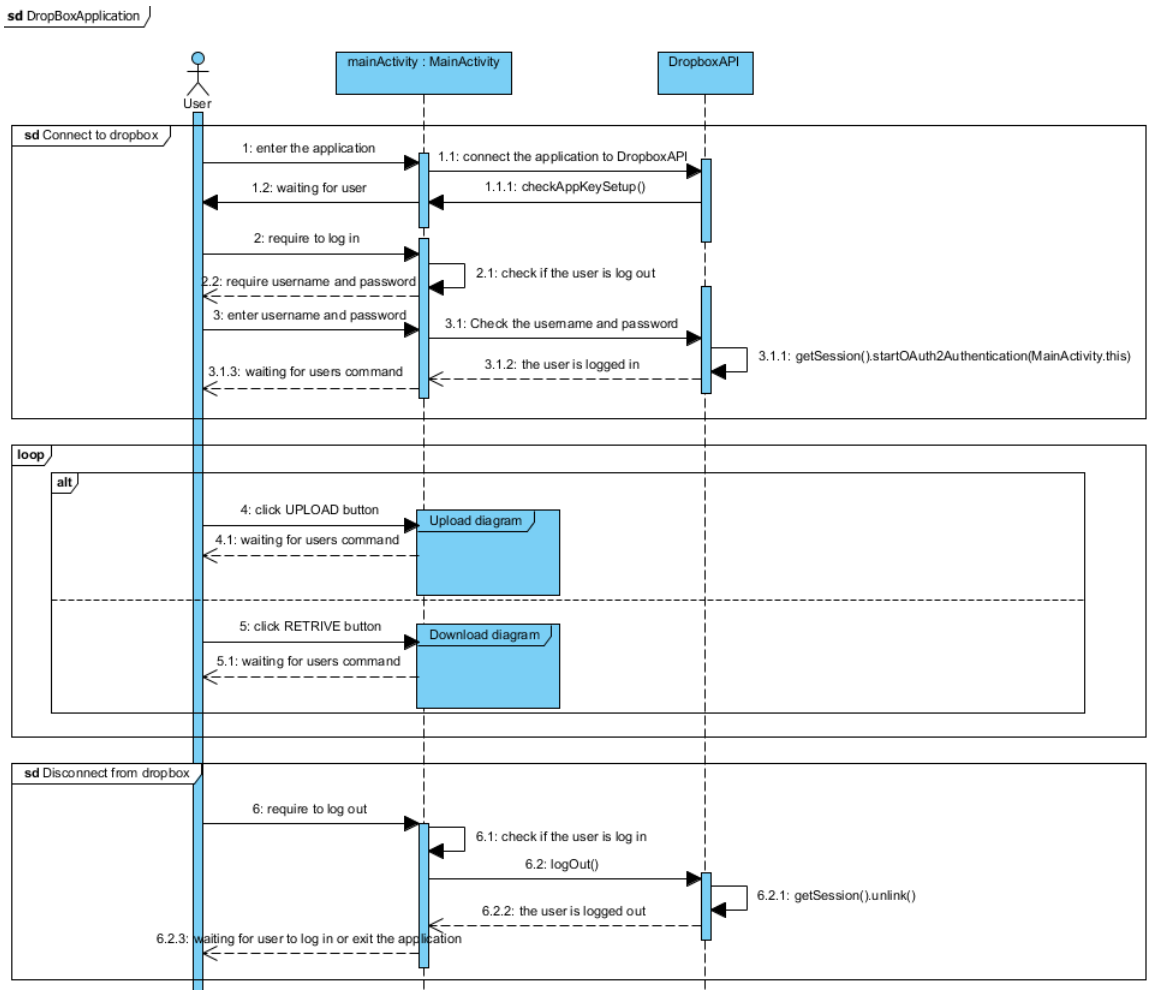


**Figure 4.2:** UML Sequence Diagram for DropBoxCrypt application

I Application checks if a fingerprint is already registered in the device.

i. If not, application requires user to register his fingerprint.

ii. Fingerprint is set up.

II Application creates key and stores into Android's Key Store.

2. User requires retrieving a file from Dropbox.

I Application opens an authentication dialog which provides finger-

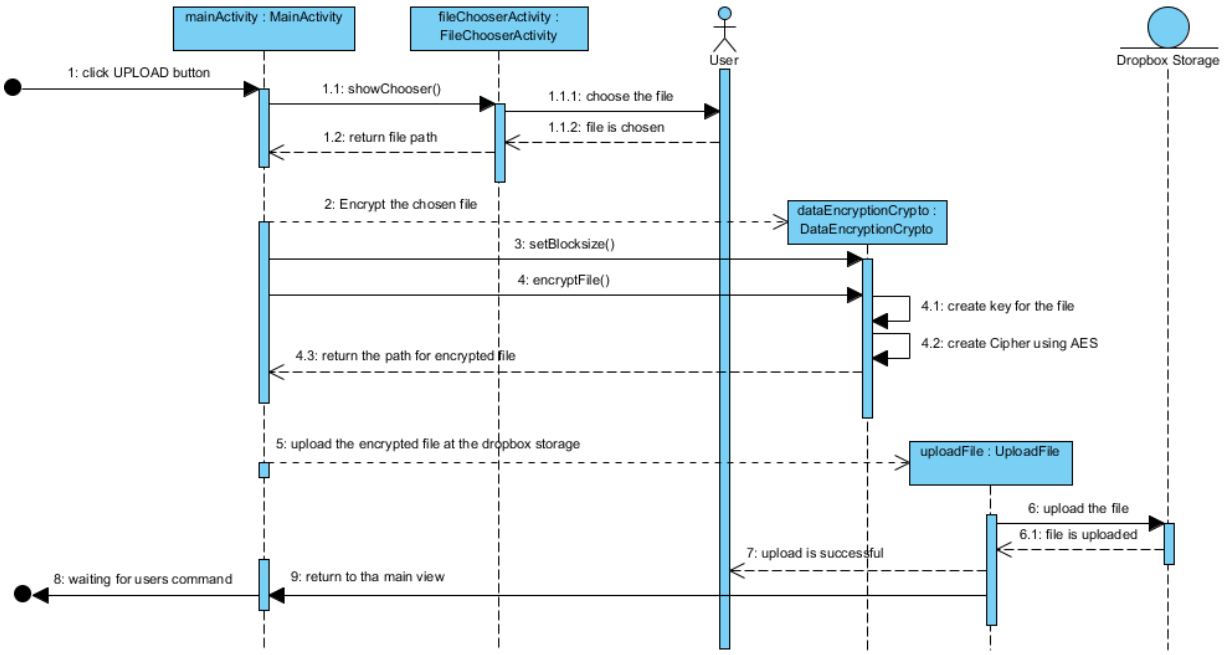


**Figure 4.3:** UML Sequence Diagram for Upload

print and password based authentications.

i. If the user scans his fingerprint, the application checks for fingerprints accessibility.

   A. If the fingerprints are accessible, application tries to match user's fingerprint to other available fingerprints.

   B. If the stored fingerprints are not accessible, application asks user to enter backup password.

ii. If the user enters system password, application proceeds to password authentication.

   A. If the password is available, application tries to authenticate the user.

II If authentication is unsuccessful, application returns back to home screen.

III If authentication is successful, application shows Dropbox file chooser.

i. User selects his desired file.

ii. Application takes targeted file's file path.

iii. Application retrieves the encrypted file.

IV Application decrypts the file.

i. Application sets up the block size.

ii. Decrypted file is available in the phone storage.

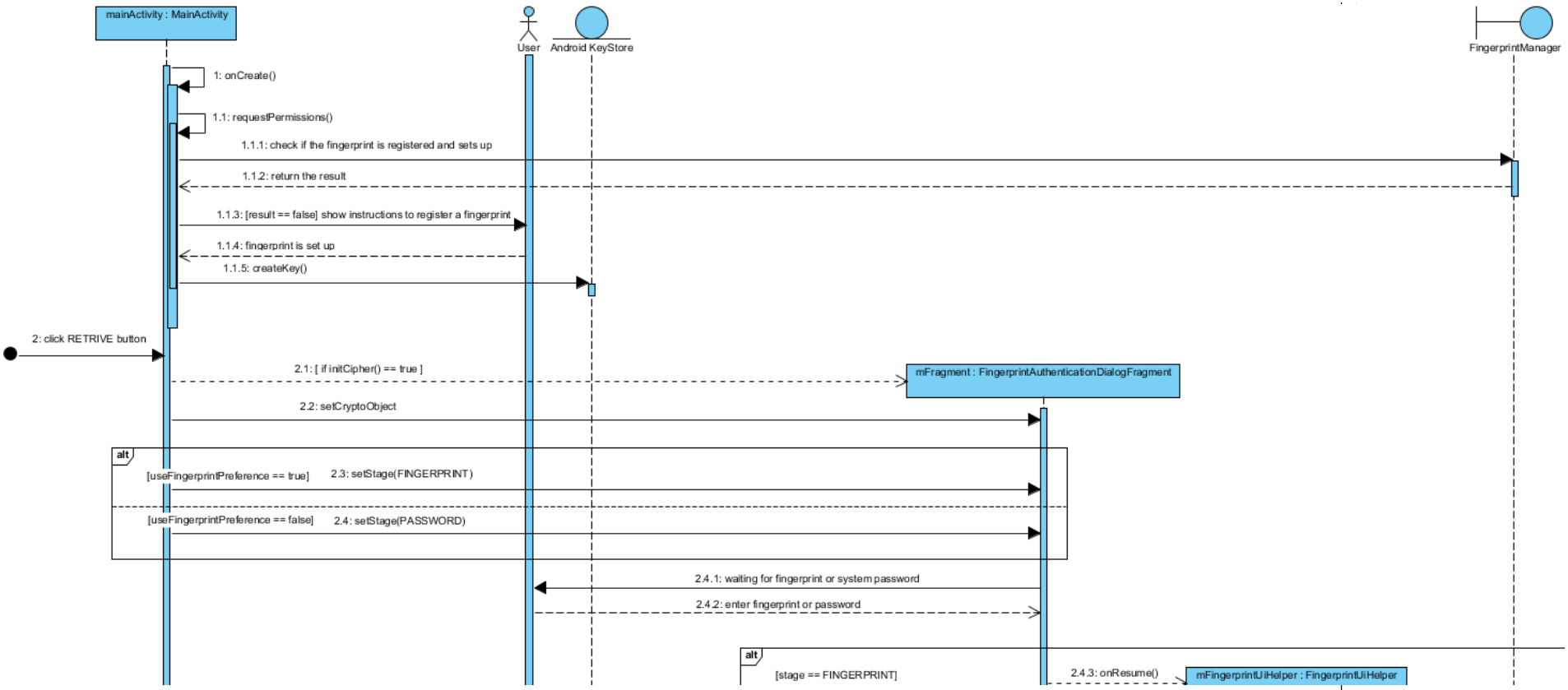


**Figure 4.4:** UML Sequence diagram for Download - I

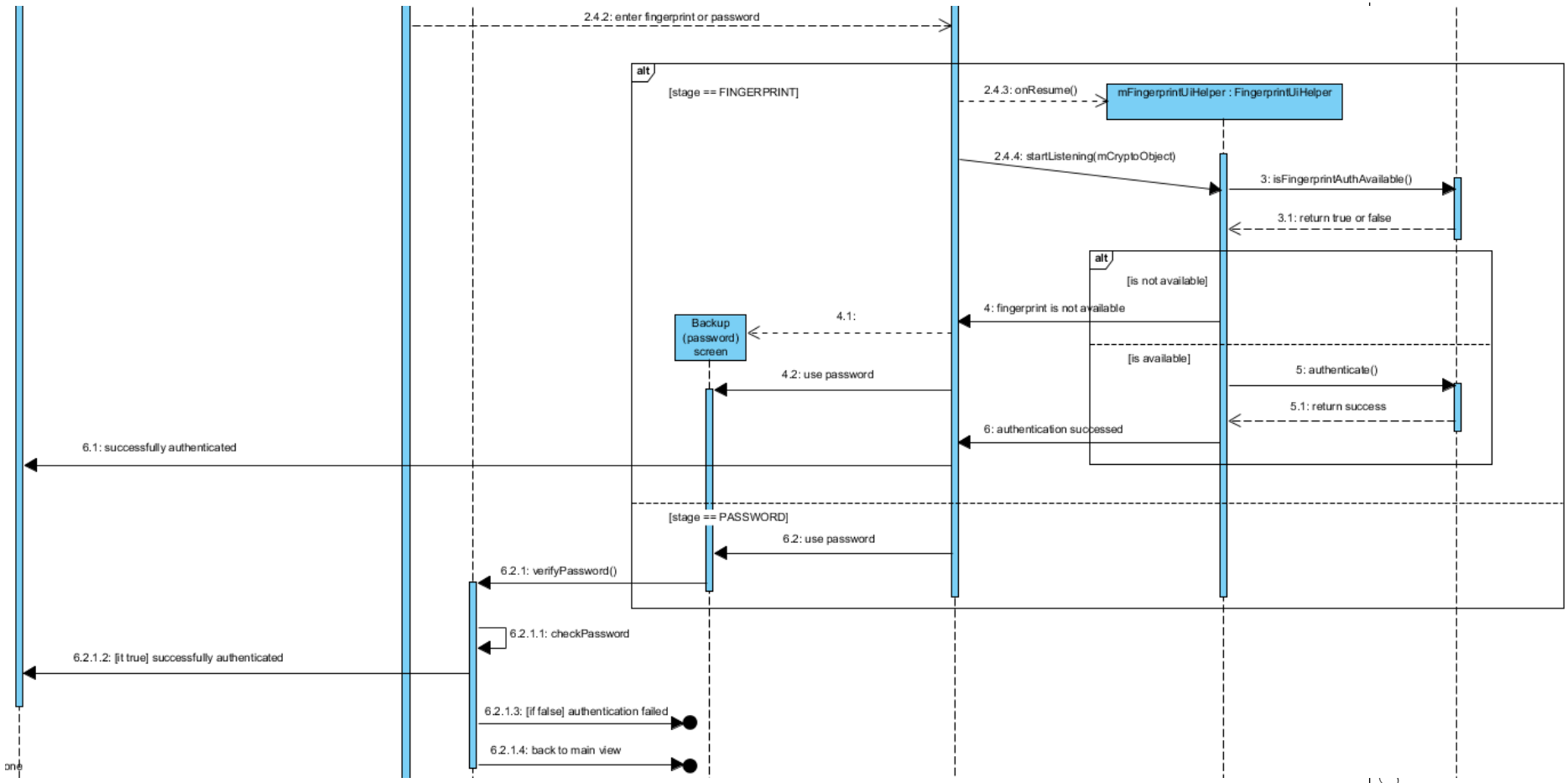


**Figure 4.5:** UML Sequence diagram for Download - II

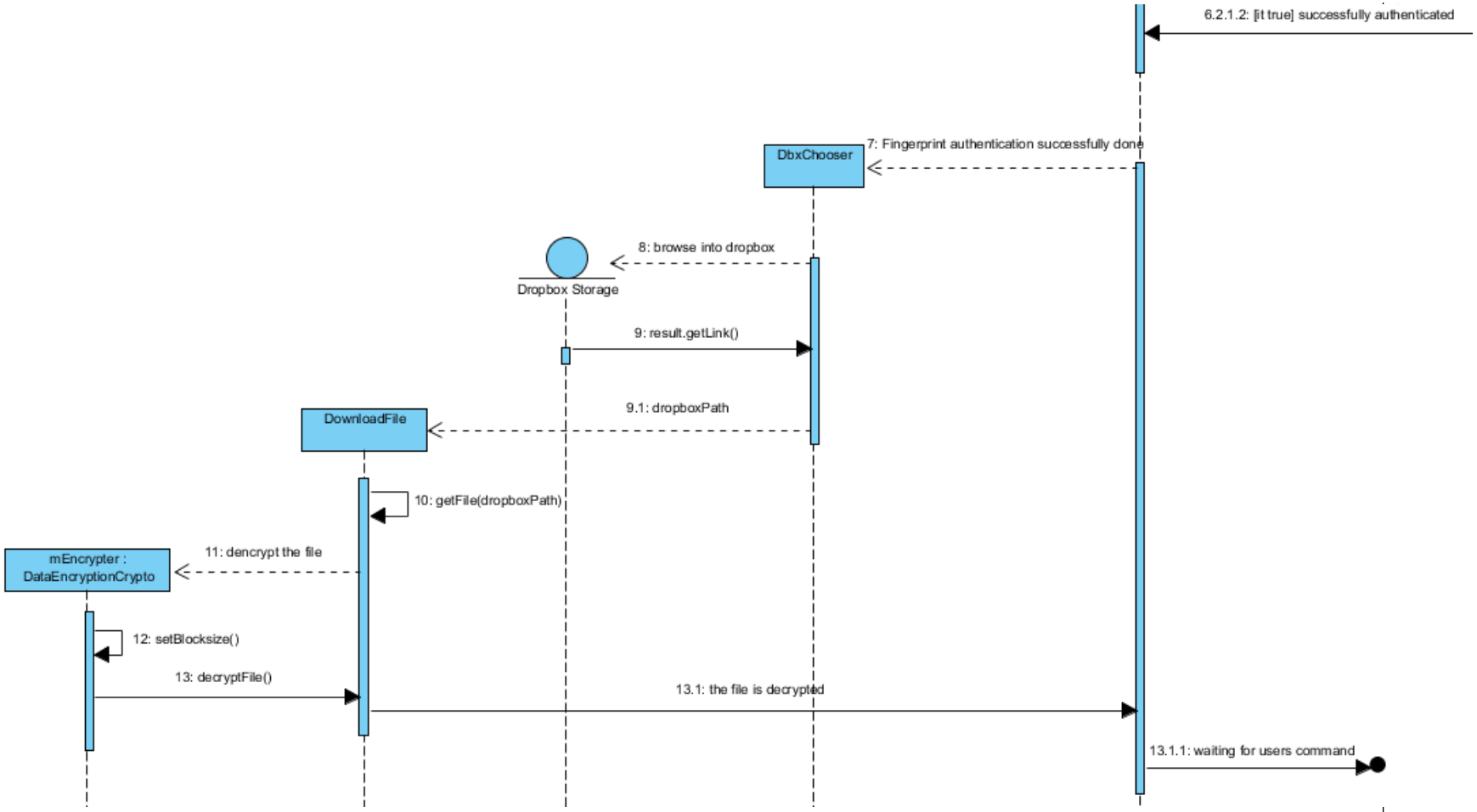


**Figure 4.6:** UML Sequence diagram for Download - III

### 4.3.3 Dropbox Security

When we start building an app on the Dropbox Platform [90] [91], we first need to obtain app permissions. For this purpose, we register a Dropbox app in the App Console [92], where we need to choose the right permission (access type) for our application. In the App Console an App key and App secret are created, and they are needed to link our application with Dropbox API. A dedicated folder named after our application is created within the Apps folder of a user's Dropbox and the content of our application is moved into this folder. We provide the app key and secret and pass both values to the new DropboxAPI object like in the Source code 4.1.

Source code 4.1: Get application permissions

```
private static final String APP_KEY = "4b0fdj4re09sy62";
private static final String APP_SECRET = "9ghy3js86qmjul0";

DropboxAPI<AndroidAuthSession> mApi;

AppKeyPair appKeyPair = new AppKeyPair(APP_KEY, APP_SECRET);
AndroidAuthSession session = new AndroidAuthSession(appKeyPair);
mApi = new DropboxAPI<AndroidAuthSession>(session);
```

Dropbox uses OAuth 2 [93] for user authentication in Dropbox and gives permission to the application to access user's data on Dropbox. Once the user proves his identity, the OAuth process returns an access token to our app and we need to send it with each subsequent API request to uniquely identify both, our application and the end user. With OAuth there is no need for storing and transmitting the user's Dropbox password, which makes OAuth a safer and more secure form of API authorization for our users. Authentication flow starts by calling the *startOAuth2Authentication()* method (Source code 4.2).

Source code 4.2: Uses of OAuth 2 for user authentication on Dropbox

```
mApi.getSession().startOAuth2Authentication(MainActivity.this);
```

To finish authentication after the user returns to the application, we need Source code 4.3.

Source code 4.3: Uses of OAuth 2 for user authentication on Dropbox

```
protected void onResume() {
   super.onResume();
   AndroidAuthSession session = mApi.getSession();
   if (session.authenticationSuccessful()) {
       try {
           session.finishAuthentication();
           storeAuth(session);
           setLoggedIn(true);
       } catch (IllegalStateException e) {
           showToast("Couldn't authenticate with Dropbox:" +
               e.getLocalizedMessage());
       }
   }
}
```

To upload a file in the Dropbox storage, we need to use the *putFile()* method. *putFile* makes a network call, so we have to make sure to invoke it on a background thread. It takes a path pointing to where we want the file in our Dropbox, an InputStream to be uploaded there, and the input's length (Source code 4.4).

Source code 4.4: Uploading a file in the Dropbox storage

```
protected Boolean doInBackground(Void... params) {
  try {
      FileInputStream fis = new FileInputStream(mFile);
      String path = mPath + mFile.getName();
      mRequest = mApi.putFile(path, fis, mFile.length(),
```

To download a file we use the opposite method *getFile()* (Source code 4.5).

Source code 4.5: Downloading a file from the Dropbox storage

```
protected Boolean doInBackground(Void... params) {
  File cachePath =
      Environment.getExternalStoragePublicDirectory(Environment
                  .DIRECTORY_DOWNLOADS);
  try {
      File file = new File(String.valueOf(cachePath) + mNameFile);
      mFos = new FileOutputStream(file);
      DropboxAPI.DropboxFileInfo info = mApi.getFile(mPath +
          mNameFile, null, mFos, null);
```

### 4.3.4 Android encryption

Android encryption is provided by Android Cryptography APIs. It consists of three main packages [94]:

**javax.crypto:** This package provides the classes and interfaces for cryptographic applications implementing algorithms for encryption, decryption, or key agreement.

**javax.crypto.interfaces:** This package provides the interfaces needed to implement the key agreement algorithm.

**javax.crypto.spec:** This package provides the classes and interfaces needed to specify keys and parameter for encryption.

Before encrypting the file, we must set up secret key spec for 128-bit AES encryption and decryption (Source code 4.6).

Source code 4.6: Setting up secret key for encryption and decryption

```
public static SecretKey generateKey() throws
    NoSuchAlgorithmException
   {
```

```
    KeyGenerator keyGenerator = KeyGenerator.getInstance("AES");
    keyGenerator.init(128);
    SecretKey key = keyGenerator.generateKey();
    return key;
  }
```

Next, we create *Cipher* to encode the original data with AES (Source code 4.7).

Source code 4.7: Encrypting the original data

```
FileOutputStream encfos = new FileOutputStream(outfile);

Cipher encipher;
encipher = Cipher.getInstance("AES");
encipher.init(Cipher.ENCRYPT_MODE, mKey);
CipherOutputStream cos = new CipherOutputStream(encfos, encipher);
```

To decrypt a file, we get the key saved in the .key file and create *Cipher* using "AES" provider (Source code 4.8).

Source code 4.8: Decrypting the encrypted file

```
SecretKey mKey = KeyStoreUtils.loadKey(tmpFile);

Cipher decipher = Cipher.getInstance("AES");
decipher.init(Cipher.DECRYPT_MODE, mKey);
CipherInputStream cis = new CipherInputStream(fis, decipher);
```

### 4.3.5 Fingerprint API

Android API 23 [95] offers to authenticate users by using their fingerprint scans, which are carefully contained within secure hardware on supported devices. This guards against malicious actors, ensuring that users use their fingerprint safely, even in untrusted applications. Android also provides pro-

tection for application developers, providing assurances that a user's fingerprint has been positively identified before providing access to secure data or resources. This API provides cryptographic-level security for both offline data and online interactions. When a user activates their fingerprint reader, they're unlocking a hardware-backed cryptographic vault. The developer can choose what type of key is stored in that vault. In our application, symmetric keys are used.

The fingerprint API is used in conjunction with the Android Keystore system [96]. This system lets us store cryptographic keys in a container to make it more difficult to extract from the device. Once keys are in the key store, they can be used for cryptographic operations with the key material remaining non-exportable.

### Using fingerprint in our application

First a symmetric key is created in the Android Keystore using *KeyGenerator* [97] which can only be used after the user has been authenticated with fingerprint and pass a *KeyGenParameterSpec* [98]. By setting *KeyGenParameterSpec.Builder.setUserAuthenticationRequired* to true, we require the user to authenticate with a fingerprint (Source code 4.9).

Source code 4.9: Create a symmetric key in the Android Keystore and require the user to authenticate with a fingerprint

```
KeyGenerator mKeyGenerator;

public void createKey() {
  try {
      mKeyStore.load(null);

      mKeyGenerator.init(new KeyGenParameterSpec.Builder(KEY_NAME,
              KeyProperties.PURPOSE_ENCRYPT |
                  KeyProperties.PURPOSE_DECRYPT)
             .setBlockModes(KeyProperties.BLOCK_MODE_CBC)
```

```
            .setUserAuthenticationRequired(true)
            .setEncryptionPaddings(KeyProperties
                                   .ENCRYPTION_PADDING_PKCS7)
            .build());
        mKeyGenerator.generateKey();
    } catch (NoSuchAlgorithmException |
        InvalidAlgorithmParameterException | CertificateException |
        IOException e) {
        throw new RuntimeException(e);
    }
}
```

Authenticating the users via a fingerprint scan is done with an instance of the new *FingerprintManager* class [99]. *FingerprintManager* is a class that coordinates access to the fingerprint hardware. The start of listening to a fingerprint on the fingerprint sensor is done by calling *FingerprintManager.authenticate()* with a Cipher initialized with the symmetric key. Once the fingerprint (or password) is verified, the *FingerprintManager.AuthenticationCallback #onAuthenticationSucceeded()* callback is called (Source code 4.10).

Source code 4.10: Authenticating the users with a fingerprint scan

```
private boolean initCipher() {
    try {
        mKeyStore.load(null);
        SecretKey key = (SecretKey) mKeyStore.getKey(KEY_NAME, null);
        mCipher.init(Cipher.ENCRYPT_MODE, key);
        return true;
    } catch (KeyPermanentlyInvalidatedException e) {
        return false;
    } catch (KeyStoreException | CertificateException |
        UnrecoverableKeyException | IOException |
        NoSuchAlgorithmException | InvalidKeyException e) {
```

```
      throw new RuntimeException("Failed to init Cipher", e);
  }
}

mRetrieve.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View v) {
   if (initCipher()) {
       mFragment.setCryptoObject(new
           FingerprintManager.CryptoObject(mCipher));
       boolean useFingerprintPreference =
           mSharedPreferences.getBoolean(getString(R.string
                       .use_fingerprint_to_authenticate_key), true);
       if (useFingerprintPreference) {
           mFragment.setStage(FingerprintAuthenticationDialogFragment
                       .Stage.FINGERPRINT);
       } else {
           mFragment.setStage(FingerprintAuthenticationDialogFragment
                       .Stage.PASSWORD);
       }
       mFragment.show(getFragmentManager(), DIALOG_FRAGMENT_TAG);
   } else {
       mFragment.setCryptoObject(new
           FingerprintManager.CryptoObject(mCipher));
       mFragment.setStage(FingerprintAuthenticationDialogFragment
                       .Stage.NEW_FINGERPRINT_ENROLLED);
       mFragment.show(getFragmentManager(), DIALOG_FRAGMENT_TAG);
    }
  }
});
```

```
public void startListening(FingerprintManager.CryptoObject
    cryptoObject) {
        if (!isFingerprintAuthAvailable()) {
            return;
        }
        mCancellationSignal = new CancellationSignal();
        mSelfCancelled = false;
        mFingerprintManager.authenticate(cryptoObject,
            mCancellationSignal, 0 /* flags */, this, null);
        mIcon.setImageResource(R.drawable.ic_fp_40px);
    }
```

To use fingerprint authentication in our application, we must add the **USE_FINGERPRINT** permission in the android manifest (Source code 4.11).

Source code 4.11: Fingerprint permission in the android manifest

```
<uses-permission
    android:name="android.permission.USE_FINGERPRINT"/>
```

# Chapter 5

# Conclusion and future work

Cloud computing is based on technologies such as virtualization, distributed computing, utility computing, network and software services. Cloud computing is one of today's hottest research areas, because of its economic benefits. Unfortunately, there are still some challenges to be solved. The flexibility provided by cloud computing can be both a friend and a foe from a security point of view. With the development and application of cloud computing, cloud security becomes more and more important.

In this masters thesis, we first discuss security issues for a cloud according to Cloud Security Alliance (CSA), using its Security Guidance [13] and security issues related with the cloud computing service delivery models (SPI model) [26]. These issues include security issues related to all aspects of infrastructure, including network level, host level and application level, the same as CSA's fourteen domains of cloud computing security. With the Security Guidance, we indicate that the management of security risk not only involves the technology itself, but also involves the cloud service providers, the users and the legal aspects of the data and services being used. The main goal in cloud security is to securely store and manage sensitive data that is not controlled by the owner of the data. The problem of keeping data secure and confidential is shown through the Data Security Lifecycle, where the process is divided into six stages: create, store, use, share, archive and

destroy.

One approach to protect cloud data is cloud cryptography. The first level of security where cryptography can help is secure storage, but the handicap is that we cannot outsource the processing of this data without decrypting it before. This thesis covers a few cryptographic schemes for cloud storage, and presents new cryptographic techniques that provide data security and mechanisms for searching or processing encrypted data, such as searchable encryption and attribute-based encryption. Searchable encryption [100] is a broad concept that allows the user not only to retrieve information privately but also to search it. Another approach for searchable encryption is to use asymmetric encryption. The first scheme that makes use of public key cryptography is the Public-Key Encryption with keyword Search (PEKS) scheme, proposed by Boneh [75]. By using public key cryptography it is possible for multiple persons to encrypt data but only the owner of the private key will be able to search for a keyword. More advanced solutions also allow searching with wildcards [78].

In our work, we propose a new application for securing sensitive data on Android mobile devices, with adding biometrics features to the algorithm. In our opinion, user authentication with fingerprint scanning can improve the security for sensitive data in the case of storing data into Dropbox cloud. Android devices can support up to three different fingerprint scans, which allow multiple users for this data. The problem with this feature is the impossibility of sharing the encrypted data with another remote user. We hope that with the future updates of Android system, this problem would be solved. In our application, data encryption is made by AES method, which can be the most safe, trusted technique to provide security to data in clouds compared to other available security techniques in cloud computing. Although encryption protects our data from unauthorized access, it does nothing to prevent data loss, which means to lose the keys that provide access to data.

It seems like there are enough opportunities to secure our information in

a cloud, but for further research there still remain many open problems. For data security and privacy issues, the fundamental challenges are separation of sensitive data and access control. Authorization and access control mechanisms should achieve a unified, reusable and scalable access control model. Another domain of future interest can be solving problems of cloud metadata and encryption key management.

Recent advances in cryptography could mean that future cloud computing services will be able to search, retrieve and store information in the cloud, without first decrypting it. Over the last few years, a few encryption solutions have been proposed for this purpose and as we have shown in the thesis, most cryptographic primitives are ready to be used. Thus, in the near future it is expected from cloud providers to implement them or produce efficient implementations that could ease its inclusion in open source platforms.

# Bibliography


[1] P. Mell, T. Grance, The nist definition of cloud computing.

[2] Ibm cloud.
URL http://www.ibm.com/cloud-computing/us/en/what-is-cloud-computing.html

[3] I. Gartner, Gartner it glossary, Technology Research.

[4] T. Mather, S. Kumaraswamy, S. Latif, Cloud security and privacy: an enterprise perspective on risks and compliance, " O'Reilly Media, Inc.", 2009.

[5] Cloud computing.
URL https://en.wikipedia.org/wiki/Cloud_computing

[6] Y. Chou, Chou's theories of cloud computing: The 5-3-2 principle (Mart 2011).
URL http://blogs.technet.com/b/yungchou/archive/2011/03/03/chou-s-theories-of-cloud-computing-the-5-3-2-principle.aspx

[7] Multitenancy (and resource pooling).
URL http://whatiscloud.com/cloud_characteristics/multi_tenancy

[8] F. Ogigau-Neamtiu, Cloud computing security issues, Journal of Defense Resources Management 3 (2) (2012) 141.

[9] W. A. Jansen, Cloud hooks: Security and privacy issues in cloud computing, in: System Sciences (HICSS), 2011 44th Hawaii International Conference on, IEEE, 2011, pp. 1–10.

[10] R. D. Caytiles, S. Lee, B. Park, Cloud computing: The next computing paradigm, International Journal of Multimedia and Ubiquitous Engineering 7 (2012) 297–302.

[11] Utility computing.
URL http://www.techopedia.com/definition/14622/utility-computing

[12] [link].
URL https://cloudsecurityalliance.org/

[13] G. Brunette, R. Mogull, et al., Security guidance for critical areas of focus in cloud computing v2. 1, Cloud Security Alliance (2009) 1–76.

[14] V. C. Bhagawat, A. A. L. Kumar, Survey on data security issues in cloud environment.

[15] Security is the top concern for public cloud, but what does that really mean?
URL http://blogs.gartner.com/neil_macdonald/2010/12/16/security-is-the-top-concern-for-public-cloud-but-what-does-that-really-mean/

[16] N. Gohring, Amazon's data storage service hit by outage, 2008.
URL http://www.computerworld.com/article/2537118/data-center/amazon-s-data-storage-service-hit-by-outage.html

[17] R. Kay, 'the internet is down.' what does that really mean?, 2009.
URL http://www.computerworld.com/article/2523959/networking/-the-internet-is-down---what-does-that-really-mean-.html

[18] Google docs leaks out private data, 2009.
URL http://www.infosecurity-magazine.com/news/google-docs-leaks-out-private-data/

[19] J. Kirk, Google's gmail hit with two-hour outage, 2009.
URL http://www.computerworld.com/article/2531439/networking/google-s-gmail-hit-with-two-hour-outage.html

[20] D. Chen, H. Zhao, Data security and privacy protection issues in cloud computing, in: Computer Science and Electronics Engineering (ICCSEE), 2012 International Conference on, Vol. 1, IEEE, 2012, pp. 647–651.

[21] G. Clarke, Microsoft's azure cloud suffers first crash, 2009.
URL http://www.theregister.co.uk/2009/03/16/azure_cloud_crash/

[22] J. Brodkin, Loss of customer data spurs closure of online storage service 'the linkup', 2008.
URL http://www.networkworld.com/article/2274737/data-center/loss-of-customer-data-spurs-closure-of-online-storage-servic
html

[23] Cloud computing security.
URL http://en.wikipedia.org/wiki/Cloud_computing_security

[24] Seven steps for building security in the cloud from the ground up, in: Planning Guide cloud security, Intel IT Center, 2012.

[25] C. Alliance, Security guidance for critical areas of focus in cloud computing v3. 0, Cloud Security Alliance.

[26] S. Subashini, V. Kavitha, A survey on security issues in service delivery models of cloud computing, Journal of network and computer applications 34 (1) (2011) 1–11.

[27] J. Brodkin, Gartner: Seven cloud-computing security risks, Infoworld 2008 (2008) 1–3.

[28] C. Wang, Cloud security front and center, Retrieved March 2 (2009) 2012.

[29] M. Almorsy, J. Grundy, I. Müller, et al., An analysis of the cloud computing security problem, in: Proceedings of APSEC 2010 Cloud Workshop, Sydney, Australia, 30th Nov, 2010.

[30] J. Forum, Cloud cube model: Selecting cloud formations for secure collaboration (April 2009).
URL `https://collaboration.opengroup.org/jericho/cloud_cube_model_v1.0.pdf`

[31] Oesd better polices for better lives.
URL `http://www.oecd.org/`

[32] A. P. E. Cooperation, APEC privacy framework, Asia Pacific Economic Cooperation Secretariat, 2005.

[33] E. Directive, 95/46/ec of the european parliament and of the council of 24 october 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data, Official Journal of the EC 23 (6).

[34] E. Parliament, Directive 2002/58/ec of the european parliament and of the council, Official Journal JL 201 (2002) 0037 – 0047.

[35] U. Government, Gramm-leach-bliley act, in: Public Law 106-102, 1999.

[36] A. Act, Health insurance portability and accountability act of 1996, Public Law 104-191 104 (1996) 191.

[37] F. T. Commission, et al., Children's online privacy protection act of 1998, Public Law 105-277.

[38] Governance, risk management, and compliance, Gartner.
URL https://en.wikipedia.org/wiki/Governance,_risk_management,_and_compliance

[39] Data security lifecycle 2.0 (September 2011).
URL https://www.securosis.com/blog/data-security-lifecycle-2.0

[40] P. Cichonski, T. Millar, T. Grance, K. Scarfone, Computer security incident handling guide, NIST Special Publication 800 (2012) 61.

[41] S. working Group, et al., defined categories of service 2011 (2011).

[42] Daemonic dispatches.
URL http://www.daemonology.net/blog/2008-12-18-AWS-signature-version-1-is-insecure.html

[43] Y. Rekhter, T. Li, S. Hares, A border gateway protocol 4 (bgp-4), 2006.
URL https://tools.ietf.org/html/rfc4271

[44] P. Boothe, J. Hiebert, R. Bush, Short-lived prefix hijacking on the internet, In Proc. of the NANOG 36.

[45] P. Mockapetris, Rfc 1034: Domain names: concepts and facilities (november 1987), 2003.
URL https://www.ietf.org/rfc/rfc1034.txt

[46] M. Zalewski, Browser security handbook, Google Code.

[47] Sans.
URL https://www.sans.org/about/

[48] Cis critical security controls.
URL https://www.sans.org/critical-security-controls/

[49] Top 10 2007.
URL https://www.owasp.org/index.php/Top_10_2007

[50] Owasp top ten project.
URL https://www.owasp.org/index.php/OWASP_Top_10#tab=OWASP_Top_10_for_2013

[51] S. VivinSandar, S. Shenai, Economic denial of sustainability (edos) in cloud services using http and xml based ddos attacks, International Journal of Computer Applications (0975 – 8887) 41 (20) (2012) 11–16.

[52] K. Rauber, Cloud cryptography, International Journal of Pure and Applied Mathematics 85 (1) (2013) 1–11.

[53] B. Prince, Ibm discovers encryption scheme that could improve cloud security, spam filtering, E-Week. com.

[54] D. Boneh, E.-J. Goh, K. Nissim, Evaluating 2-dnf formulas on ciphertexts, in: Theory of cryptography, Springer, 2005, pp. 325–341.

[55] B. Lewis, Data lineage: The next generation (August 2008).
URL http://tdan.com/data-lineage-the-next-generation/8151

[56] Y. L. Simmhan, B. Plale, D. Gannon, A survey of data provenance techniques, Computer Science Department, Indiana University, Bloomington IN 47405.

[57] Wikipedia, Data remanence.
URL https://en.wikipedia.org/wiki/Data_remanence

[58] J. Bloomberg, Data remanence: Cloud computing shell game (May 2011).
URL http://zapthink.com/2011/05/19/data-remanence-cloud-computing-shell-game/

[59] P. Yong, Z. Wei, X. Feng, Z.-h. DAI, G. Yang, D.-q. CHEN, Secure cloud storage based on cryptographic techniques, The Journal of China Universities of Posts and Telecommunications 19 (2012) 182–189.

[60] S. Kamara, K. Lauter, Cryptographic cloud storage, in: Financial Cryptography and Data Security, Springer, 2010, pp. 136–149.

[61] M. Barua, X. Liang, R. Lu, X. Shen, Espac: Enabling security and patient-centric access control for ehealth in cloud computing, International Journal of Security and Networks 6 (2-3) (2011) 67–76.

[62] P. Mockapetris, Rfc 1034: Domain names-concepts and facilities, 1987. URL `https://www.ietf.org/rfc/rfc1034.txt`

[63] S. Zarandioon, D. D. Yao, V. Ganapathy, K2c: Cryptographic cloud storage with lazy revocation and anonymous access, in: Security and Privacy in Communication Networks, Springer, 2011, pp. 59–76.

[64] J. Somorovsky, C. Meyer, T. Tran, M. Sbeiti, J. Schwenk, C. Wietfeld, Sec2: Secure mobile solution for distributed public cloud storages., in: CLOSER, 2012, pp. 555–561.

[65] R. A. Popa, J. R. Lorch, D. Molnar, H. J. Wang, L. Zhuang, Enabling security in cloud storage slas with cloudproof., in: USENIX Annual Technical Conference, Vol. 242, 2011.

[66] S. Ruj, A. Nayak, I. Stojmenovic, Dacc: Distributed access control in clouds, in: Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on, IEEE, 2011, pp. 91–98.

[67] A. Kiayias, Y. Tsiounis, M. Yung, Group encryption, in: Advances in Cryptology–ASIACRYPT 2007, Springer, 2007, pp. 181–199.

[68] J. Feng, Y. Chen, D. H. Summerville, A fair multi-party non-repudiation scheme for storage clouds, in: Collaboration Technologies and Systems (CTS), 2011 International Conference on, IEEE, 2011, pp. 457–465.

[69] J. Feng, Y. Chen, D. Summerville, W.-S. Ku, Z. Su, Enhancing cloud storage security against roll-back attacks with a new fair multi-party non-repudiation protocol, in: Consumer Communications and Networking Conference (CCNC), 2011 IEEE, IEEE, 2011, pp. 521–522.

[70] S. Kamara, C. Papamanthou, T. Roeder, Cs2: a semantic cryptographic cloud storage system, 2011.

[71] S. S. Chow, C.-K. Chu, X. Huang, J. Zhou, R. H. Deng, Dynamic secure cloud storage with provenance, in: Cryptography and Security: From Theory to Applications, Springer, 2012, pp. 442–464.

[72] D. X. Song, D. Wagner, A. Perrig, Practical techniques for searches on encrypted data, in: Security and Privacy, 2000. S&P 2000. Proceedings. 2000 IEEE Symposium on, IEEE, 2000, pp. 44–55.

[73] R. Curtmola, J. Garay, S. Kamara, R. Ostrovsky, Searchable symmetric encryption: improved definitions and efficient constructions, in: Proceedings of the 13th ACM conference on Computer and communications security, ACM, 2006, pp. 79–88.

[74] E.-J. Goh, et al., Secure indexes., Vol. 2003, 2003, p. 216.

[75] D. Boneh, G. Di Crescenzo, R. Ostrovsky, G. Persiano, Public key encryption with keyword search, in: Advances in Cryptology-Eurocrypt 2004, Springer, 2004, pp. 506–522.

[76] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, H. Shi, Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions, in: Advances in Cryptology–CRYPTO 2005, Springer, 2005, pp. 205–222.

[77] D. J. Park, K. Kim, P. J. Lee, Public key encryption with conjunctive field keyword search, in: Information security applications, Springer, 2004, pp. 73–86.

[78] D. Boneh, B. Waters, Conjunctive, subset, and range queries on encrypted data, in: Theory of cryptography, Springer, 2007, pp. 535–554.

[79] M. Bellare, A. Boldyreva, A. O'Neill, Deterministic and efficiently searchable encryption, in: Advances in Cryptology-CRYPTO 2007, Springer, 2007, pp. 535–552.

[80] A. Sahai, B. Waters, Fuzzy identity-based encryption, in: Advances in Cryptology–EUROCRYPT 2005, Springer, 2005, pp. 457–473.

[81] Introduction to attribute based encryption (abe).
URL `http://gleamly.com/article/introduction-attribute-based-encryption-a`

[82] V. Goyal, O. Pandey, A. Sahai, B. Waters, Attribute-based encryption for fine-grained access control of encrypted data, in: Proceedings of the 13th ACM conference on Computer and communications security, Acm, 2006, pp. 89–98.

[83] J. Bethencourt, A. Sahai, B. Waters, Ciphertext-policy attribute-based encryption, in: Security and Privacy, 2007. SP'07. IEEE Symposium on, IEEE, 2007, pp. 321–334.

[84] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, D. Song, Provable data possession at untrusted stores, in: Proceedings of the 14th ACM conference on Computer and communications security, Acm, 2007, pp. 598–609.

[85] A. Juels, B. S. Kaliski Jr, Pors: Proofs of retrievability for large files, in: Proceedings of the 14th ACM conference on Computer and communications security, Acm, 2007, pp. 584–597.

[86] H. Shacham, B. Waters, Compact proofs of retrievability, Vol. 26, Springer, 2013, pp. 442–483.

[87] G. Ateniese, S. Kamara, J. Katz, Proofs of storage from homomorphic identification protocols, in: Advances in Cryptology–ASIACRYPT 2009, Springer, 2009, pp. 319–333.

[88] C. C. Erway, A. Küpçü, C. Papamanthou, R. Tamassia, Dynamic provable data possession, Vol. 17, Acm, 2015, p. 15.

[89] Chooser.
URL https://www.dropbox.com/developers/chooser

[90] Core api.
URL https://www.dropbox.com/developers-v1/core/docs

[91] Dropbox platform developer guide.
URL https://www.dropbox.com/developers/reference/developer-guide

[92] [link].
URL https://www.dropbox.com/developers/apps

[93] Oauth guide.
URL https://www.dropbox.com/developers/reference/oauth-guide

[94] C. Liu, Android encryption with the android cryptography api (2013).
URL http://www.developer.com/ws/android/encrypting-with-android-cryptography-api.html

[95] Android 6.0 apis.
URL http://developer.android.com/about/versions/marshmallow/android-6.0.html

[96] Android keystore system.
URL http://developer.android.com/training/articles/keystore.html#SecurityFeatures

[97] Keygenerator.
URL `https://developer.android.com/reference/javax/crypto/KeyGenerator.html`

[98] Keygenparameterspec.
URL `https://developer.android.com/reference/android/security/keystore/KeyGenParameterSpec.html`

[99] Fingerprintmanager.
URL `http://developer.android.com/reference/android/hardware/fingerprint/FingerprintManager.html`

[100] I. Agudo, D. Nuñez, G. Giammatteo, P. Rizomiliotis, C. Lambrinoudakis, Cryptography goes to the cloud, in: Secure and Trust Computing, Data Management, and Applications, Springer, 2011, pp. 190–197.